# LINUX/UNIX – HIGH PERFORMANCE COMPUTING

## 1 Einleitung

**UNIX Story**: ursprünglich als wissenschaftliches Softwareentwicklungssystem konzipiert, vorwiegend an Universitäten eingesetzt worden, Bedeutung gewann aufgrund zunehmender Heterogenität von Computersystemen und Vernetzung
**Kommerzielles UNIX**: z.B. **AIX** (Advanced Interactive Executive), **SunOS**

### 1.1. Einordnung und Eigenschaften von UNIX:

#### 1.1.1. Rolle und Aufgaben eines Betriebssystem:

**Computersystem**: Einheit aus Hard- und Software
**Betriebssystem**: Betriebssoftware mit weiteren Systemprogrammen, Rechnerumgebung, in der Anwendungsprogramme entwickelt und ausgeführt werden, übernimmt vorrangig Verwaltungs- und Dienstfunktionen (Bedürfnis-anpassungsfähig, Steuerung, Protokollierung, Ressourcenverwaltung, Sicherheit und Stabilität), Bindeglied zwischen Hardware und Benutzer
**Anwendungssoftware**: Applikationsprogramme, die auf dem Betriebssystem aufsetzen

#### 1.1.2. Benutzung eines Betriebssystem;

- **Über Benutzerschnittstelle**: Kommandoeingabe über Konsole, modern über grafische Oberfläche (**GUI**)
- **Über Programmierschnittstelle**: Verbindung zu einem Programmiersystem (Compiler, Bibliotheken) zum Aufruf von Systemdiensten aus Programmen

#### 1.1.3. Klassifikation von Betriebssystemen:

1. zeitliches Verhalten bei der Kommunikation mit der "Umgebung" (Betriebsart):
   - **Job-Systeme, Stapelverarbeitung**: Antwortzeit des OS praktisch nicht begrenzt
   - **Dialogsysteme (interaktive Systeme)**: Antwortzeit im "Mittel" unterhalb eines Grenzwertes
   - **Echtzeitsysteme**: Einhaltung eines Grenzwertes für Antwortzeit unbedingt erforderlich
2. Anzahl der (parallel) bedienbaren Benutzer:
   - **Ein-Nutzer-Betriebssysteme (single user)**
   - **Mehr-Nutzer-Betriebssysteme (multi user)**
3. Anzahl der parallel bearbeitbaren Nutzeraufträge:
   - **Ein-Prozeß-Betriebssystem (single-tasking)**: nur ein Auftrag gleichzeitig bearbeitbar, **z.B. MS DOS**
   - **Mehr-Prozeß-Betriebssystem (multi-tasking)**: mehrere Aufträge gleichzeitig bearbeitbar, **z.B. UNIX**, verfeinerte Realisierung heißt **multi-threading**

#### 1.1.4. Eigenschaften von UNIX:

→ hinsichtlich der Konzeption von Architektur und Implementierung:

- **Portabilität**: UNIX fast vollständig in **C** geschrieben, auf verschiedene Computersysteme (Hardware-Plattformen) ohne Verhaltensabweichungen übertragbar und lauffähig
- **Einfache Benutzerschnittstelle**
- **Flexibilität**: UNIX als Bau- oder Werkzeugkasten, Tools und Möglichkeiten zur Schaffung komplexer Lösungen, programmierbare Shells erlauben Ausreizung von Nuancen
- **Einheitliche Behandlung von Dateien und peripheren Geräten**: nur ein fortlaufender Datenstrom
- **Time-Sharing**: für **Multiuser**- und **Multitaskingbetrieb** notwendige Zuteilung der CPU an jeweils laufbereite Prozesse mit Prioritäten und Zeitscheiben

    - **Multitasking**: **präemptives Multitasking** (Programm hat Ausführungsgarantie, Gleichberechtigung von Programmen), **kooperatives Multitasking** (DOS / WIN 3.x, Programm läuft bis totale Ausführung beendet oder freiwillig beendet wird, Ungleichberechtigung)
    - **Multiuser**: Gleichzeitige Arbeit von selben oder verschiedenen virtuellen Terminals aus
- **Netzwerkfähigkeiten**: Netzanbindung auch für Systemadministratoren und Techniker über Zugriffsfenster, effektive Wartung
- **LINUX Standardisierung**: durch internationale Gremien, offenes herstellerunabhängiges System

## 1.1.5. Architektur von UNIX:

→ kernbasierte Form (**kernel**) mit darauf aufbauender Schale (**shell**)
**Kernel**: monolithisches Gebilde, bietet grundlegende OS Funktionen und dafür benötigte Datenstrukturen (Prozeßverwaltung, Dateisystem, Gerätetreiber), Herz oder Gehirn von LINUX
**Shell**: Verbindung zwischen Nutzer und Kernel, **Kommandointerpreter** (Zerlegung in geeignete Folge elementarer Funktionen), Ausführungsbenachrichtigung an Kernel mit **Systemrufen** (**system calls**), **z.B. C-, Bourne- oder BASH Shell** (unterscheiden sich in der Syntax)
**Architektur**: **Shell umhüllt → Kern umhüllt → Hardware**

## 1.2. Historie und aktuelle Standardisierung:

**→ OS Vorprodukt MULTICS**
**60er Jahre**: von **AT&T (American Telephone and Telegraph) Bell Laboratories** und dem **MIT** (**Masachusetts Institute of Technology**), läuft zuerst auf **DEC PDP-7** Rechnern, dann Portierung
**80er Jahre**: Zersplittung der Entwicklungen durch Verbreitung des Quellcode

- **XENIX** (Microsoft): Portierung auf PC, ging ins UNIX System V ein
- **BSD UNIX** (Berkeley System Distribution): an der kalifornischen Berkeley Universität eigenständig entstandene Version, bekannteste Version wurde von Sun Microsystems vermarktet
- **UNIX System V** (UNIX System Laboratories, USL): de-facto Standard im kommerziellen Bereich, R4 ist eine Kombination von AT&T und Sun Microsystems

**1993**: Kauf des Warenzeichen UNIX von Novell Inc., Erwerb des rechtlichen Eigentums und Verkauf von Lizenzen → schließlich Verkauf von UNIXWARE an SCO (Santa Cruz Operation)

**Standardisierungsbemühungen (Stichworte offene Systeme und Interoperabilität)**:

- **POSIX**: IEEE P1003, Definition der UNIX OS-Programmierschnittstelle für Programme in C (API – System Application Program Interface), gewährleistet Portabilität bei Benutzung definierter Funktionsaufrufe, kleinster gemeinsamer Nenner aller folgenden – teils kommerziellen – Bestrebungen
- **SVID** (System V Interface Definition von AT&T): Kommando- und Funktionsdefinition für UNIX SV
- **X/OPEN**: Vereinigung von Herstellern/Anwender, X/OPEN Portability Guide (XPG), Spezifikation für offene Systeme, Verifizierung und Zertifizierung
- **UI** (UNIX International): Firmengruppe mit Ziel Weiterentwicklung auf Basis System V R.4
- **OSF** (Open Software Foundation): Herstellervereinigung zur UNIX-OS Entwicklung und offene Entwicklungsumgebung für Anwendungen (OSF/1, OSF/Motif)

## LINUX Einführung:

**LINUX Story**: UNIX Clone, erfunden vom finnischen Informatik-Student Linus Torvalds, erste offizielle Version in **1991**, Universität Helsinki, Ziel: Ersatz für OS MINIX schaffen, Veröffentlichung von Binaries und Sourcecode, viele Programmierer perfektionieren das Betriebssystem unentgeltlich, OS für Intel-basierte PCs
**GNU**: **General Public License** (**GPL**) – Programmierarbeiten inklusive dem abgeänderten Sourcecode werden wieder der programmierenden Allgemeinheit zur Verfügung gestellt,

ermöglicht Programmentwicklern Erlangung eines legalen Copyright, man kann Source benutzen und diese modifiziert verkaufen, man muß dann jedoch den Quellcode freistellen
**Firmenunterstützung**: Siemens, IBM, Compaq, Oracle, Informix, Software AG, Sybase, Applix, Corel, Stardivision

### Renommierte modulare LINUX Software:

**4 wichtige LINUX-Softwareklassifikationen**: Console, GNOME, KDE, X11

**GUIs**: grafische Benutzeroberfläche für konsistenen Look and Feel, **KDE**, **GNOME**

**GNOME**: GNU Network Object Model Environment, Projekt: eine komplette, leicht benutzbare Desktopumgebung und ein Power Application Framework für Softwareentwickler schaffen, benutzt GTK+ als GUI Toolkit, OpenSource compliant, **www.gnome.org**

**KDE**: professionelle stabile grafische Desktopumgebung, **www.kde.org**

- **KOffice**: Suite mit K Word, K Spread, K Presenter, K Illustrator (Vektor), K Imageshop (Bitmap), Katabase, K Formula, K Chart, K Image
- **KDevelop**: C/C++ IDE (Integrated Development Environment)

**Kernel**: Herz eines UNIX Betriebssystem, Kernelarchiv: **www.kernel.org**
**X11**: X11 Programme für das X Windows System
**Xfree86**: von der Non-Profit-Organisation **Xfree86 Projcect, Inc.** frei vertreibbare Implementation des **X Windows System** (de facto industry standard), abgeleitet von der offiziellen **X11R6.3** Distribution des **X Consortium, Inc.**, für UNIX(R) und UNIX-ähnliche OS und OS/2

**Software**: in Form von Source-Code oder Paketform (bereits kompiliert)

### Dokumentation:

**Hypertextbrowser**: **Netscape Communicator** (über GUI), **lynx** (über Textkonsole)
**Info**: Infoformat mit **Infoviewer** (info) oder **Emacs** (emacs) im Infomode lesen, unter X Windows mit **tkInfo** (tkinfo) oder **Xinfo** (xinfo, ungeeignet)
**UNIX Manuals**: **Manpages** (man)
**FAQs, Howto, Readme´s**: in Unterverzeichnissen, Paket **manyfaqs**, Serie **doc**, lesbar mit **less** (liest auch gepackte Dateien)
**Freie Bücher**: Paket **books** – Serie doc (Postscript Dateien, ansehen mit Paket **gsview** – Serie gra oder Paket **gv** – Serie gra)
**Help**: **ls --help**

## SUSE LINUX Installation

**Hardwarevoraussetzungen**: **Intel 386 Prozessor** (oder kompatibel, für Grafikmodus **Pentium**), **8MB** RAM (**32MB** RAM Grafikmodus), 3,5 Zoll FDD, CD ROM, mind. **250MB** HDD für normale Konfiguration (empfohlen mind. **500MB**, Minimaleinrichtung **70MB**, Kompletteinrichtung **1GB**), Grafikkarte
**Hardwarekompatibilitätsliste**: **http://www.suse.de/cdb/D/** (anstatt www: cdb)
**Treiber**: entweder als eigenständige Module oder bereits eingebunden in den Systemkern
**Treibererkennung**: entweder über **Autoprobing** oder **Parameterübergabe** (TIP: Einstellungen aus WIN9X Installation bereithalten)

### 1. Erste LINUX Bootmöglichkeiten (BOOTSTRAPPING):

**5 Installationsvarianten:**

- **Bootdiskette**: dann Zugriff auf CDROM/HDD (→ zur Erstellung wird eine BOOT- und/oder eine ROOT-Diskette benötigt)
  - **3 Erstellungsmöglichkeiten einer Bottdiskette**:
  - o Erstellung mit **Setup.exe – Option Floppy**

- DOS-Programm **rawrite.exe** in **\dos_util**: es werden sogenannte Disk-Images (Datei-Abbilder) auf die Disketten übertragen) oder
- Unter UNIX: Formatierung: **fdformat /dev/fd0u1440**, Mounten der CD: **mount –tiso9660 /dev/cdrom /mnt**, Verzeichniswechsel: **cd /mnt/disks**, Erstellen: **dd if=<Diskette> of=/dev/fd0 bs=8192**

**Diskettentypen**:

- **BOOT-Diskette**: startet das LINUX System, enthält wichtige Gerätetreiber und eine grundlegende Version des OS
- **ROOT-Diskette**: erzeugt grundlegendes Dateisystem für den ersten Bootvorgang, wird bei **SUSE** nicht mehr benötigt

→ die **BIOS BOOT-Sequenz** ist **A,C**
→ **Inhalt der Images**: **Syslinux** (Loader, ermöglicht Kernelwahl beim Booten), **linuxrc** (lädt Module für Hardware und startet Installation)

- **CD**: bei passendem **BIOS** und CD-ROM LINUX Unterstützung
  - **Boothilfe mit Kernelparametern zur CD Erkennung**: sofern CDROM nicht gefunden: **boot: linux hd<x>=cdrom** (x: **a** – Master am 1. IDE, **b** – Slave am 1. IDE, **c** – Master am 2. IDE ...), Controllerparameter für (E)IDE Festplatten: **ide<x>=noautotune** (x: 0 – 1. IDE Controller, 1 – 2. IDE Controller ...)
- **Installation unter DOS ohne CDROM**: Kopie des Grundsystems (Kernel in **4** Dateien: 1. Installationskernel, 2. **.ikr installable kernel**, 3. **.inf** Yastbeschreibung, 4. **.map** Kernelsymbole) → Verzeichnisse anlegen (**\x, \x\suse, \x\suse\a1, \x\suse\images, \x\suse\setup, \x\suse\setup\descr**) → a1 unter WINDOWS9X/NT in **...\a1** kopieren, gewählten Kernel + sicherheitshalber **initdisk.gz** in **...\images** kopieren, **loadlin.exe** und **inst-img** in **...\setup** kopieren, alle Dateien des gleichnahmigen CD Ordners in **\descr** → Frage nach **linuxrc** Quellmedium/Device beantworten (z.B. **/dev/hda1** oder **/dev/sda1**), Quellverzeichnis **/x**
- **Netzwerk**: **NFS/FTP** oder **PLIP**, Quellmedium NETZWERK (NFS), bei Problemen evtl. **exec** Rechte des Servers mitimportieren bzw. Installationsrechner mit Namen und IP Adresse in **/etc/hosts** des Servers eintragen
- **Setup und loadlin**: **setup.exe** ruft **loadlin** auf, um Kernel für **Ur-LINUX** in Speicher zu laden, DOS in **Realmodus** (Prozessor entweder im **Realmodus** oder **virtueller 8086 Modus** mit **aktivem VCPI Server** wie z.B. **emm386.exe**) bringen, **DOS-BOX** von **OS/2** oder **NT** geht nicht! **2** alternative Startmöglichkeiten: BOOT-Disk oder loadlin direkt von CD/HDD, Verzeichnis **\loadlin** wird auto-angelegt, Start später mit **Linux.bat**, loadlin benötigt genügend RAM zum Laden des Kernels

→ **Kernel BOOT** und Start des menügeführten Installations- und Bootprogramm **linuxrc**

## 2. linuxrc

**Linuxrc**: lädt Treiber als Kernelmodule (Hardwaretreiber), lädt kleinen modularisierten Kernel und erlaubt, wichtige Module nachzuladen → Sprachauswahl → Bildschirmwahl etc.
→ **Hauptmenü** (Einstellungen, System-Information, Kernelmodule: für SCSI/PCMCIA Unterstützung wählen), EIDE und ATAPI sind fest in den Kernel eingebaut
**Weitere linuxrc Anwendungen**: Start eines autonomen, RAM-Disk basierten Rettungssystem oder Hilfe wenn ROOT Passwort vergessen

## 3. Besondere Kernelmodule über Kerneldämon nachladen (automatisch/manuell):

- **SCSI**: bei SCSI HDD und CDROM, gibt es keinen gültigen Treiber: spezielle Bootdisk mit Hardware Unterstützung benutzen (SCSI Treiber fest im Kernel eingebunden!)
- **PCMCIA**
- **Netzwerk**: Installation über NFS, FTP (siehe Netzwerk unter 1.)
- **CDROM (kein ATAPI)**:

→ SUSE liefert eine **modules** Diskette mit exotischen Treibern mit, diese muß auf Verlangen einer **linuxrc Systemmeldung** benutzt werden

→ Hardware wird in **/etc/conf.modules** spezifiziert, dort bei Problemen nachschauen, **LILO** oder **modprobe** werten diese Datei aus

## 4. Weitere linuxrc Optionen

**Alternative linuxrc Optionen**: **Installiertes System booten** (BOOT-Probleme), **Rettungssystem starten** (Reparation), **Live-CD starten** (Unsinn)
→ Quellmedium wählen und Installation starten

## 5. YAST starten:

**YAST**: **Yet Another Setup Tool**, eigentliche Installationsroutine von SUSE, Start via **Installation / System starten**,→ **Linux neu installieren**
**Alternative YAST Optionen**: **Update**, Installation im **Experten Modus**

## 6. Partitionierung:

→ Partitionierung zwingend wählen, wenn eine **Dual-Boot** Maschine aufgesetzt wird, üblicherweise wird der LINUX Bereich in **3** Teile aufgespalten

**SUSE Partitionierungsvarianten:**

- **Automatisch**: bei Erkennung freien Speicherplatzes, Vorschlag **GESAMTE PLATTE** (alle Daten gehen hier verloren, nicht empfohlen, erzeugt **2MB BOOT**-Partition, **SWAP** mit **2xRAM** Größe aber max. **128MB**, Rest als "/" **ROOT**)
- **Interaktiv**: mit **\dosutils\fips**, zwingend bei Multiboot-Systemen
- **UMSDOS**: vollständiges UNIX Dateisystem, das LINUX nutzen kann und das sich auf einer MSDOS Partition befindet, Möglichkeit ohne Repartitionierung (diese Variante benötigt eine UMSDOS-ROOT-Disk), bei Verwendung von WINDOWS sollte UMSDOS nicht verwendet werden

**Partitionen**: Festplattenaufteilung über Spezifikation der Partitionstabelle im Boot-Record (BR), in den ersten Sektoren der Festplatte angesiedelt, jede Festplatte hat genau eine Partitionstabelle mit Platz für **4** Einträge
**3 Basis-Partitionsarten**:

- **Primär**: maximal **4** pro Festplatte, zwingend notwendig für eine DOS/WINDOWS Installation
- **Erweitert**: "Container", enthält keine Daten, zur Definition logischer Partitionen
- **Logisch**: Untergliederung der erweiterten Partition, benötigt kein Platz in der Partitionstabelle im BR, Unterteilung in max. **15** SCSI Partitionen oder **63** (E)IDE Partitionen, diese enthalten Daten

→ LINUX kann auf primären oder logischen Partitionen installiert sein

**Partitionierungstools**: **fips** (LINUX), **fdisk.exe** (DOS) oder **Partition Magic** (kommerziell)
**FIPS**:

- Bei Dual Boot vorher ein **Defragmentierungstool** benutzen, freigestellter Speicher muß als **einzelner Block** am Ende der Festplatte vorliegen (Systemdateien mit den Attributen **System** oder **Hidden** können Probleme bereiten, da sie nicht verschoben werden können, also Attribute **temporär löschen**, evtl. **Windows Auslagerungsdatei deaktivieren**)
- FIPS hat Probleme mit **komprimierten** Festplatten
- FIPS über DOS/WIN Startdiskette starten
- FIPS bietet Sicherung von ROOT und BOOTSEKTOR an (**rootboot.000**)
- **Restorrb.exe** kann ursprüngliche Partitionierung nach einem FIPS Fehler wieder herstellen, sofern der ROOT und BOOTSEKTOR vorher gesichert wurden
- Unter WIN98 unbedingt **FIPS 2.0** verwenden, da eine vorhergehende Version keine **FAT32** Dateizuordnungstabellen verarbeiten kann

**Partitionsbenennung DOS versus LINUX**: alphabetische Nummerierung der Laufwerke (DOS) versus Verzeichnisnamen (UNIX/LINUX)

**Konfiguration:**
**Normale Konfiguration**: 1 primäre ROOT-Partition (restlicher verfügbarer Speicher zur Dateispeicherung), 1 SWAP-Partition (ca. **40-80MB**, Typ muß explizit gesetzt werden, ohne SWAP Partition ist LINUX lahm)
**Empfohlene Konfiguration**: **/boot** Partition für Kernel und LILO (LINUX LOADER) mit (**min. 2MB**) **5-10MB** bzw. **1 Zylinder**, "**/**" ROOT-Partition mit **1GB**, SWAP-Partition mit **64-128MB**, bei HDDs > 1,2GB bzw. <500MB gelten andere Regeln
**Professionelle Konfiguration**: abhängig von:

- **Einsatzgebiet**: **Druckserver/Router/Firewall/Internet Gateway ohne Benutzer** (386er Prozessor, 16MB SWAP, Rest für ROOT), **durchschnittliche Workstation** (BOOT 5-10MB, ROOT 180MB, min. 64MB SWAP, /home 100MB, /usr REST, /opt), **Fileserver** (extra HDD für \home), **Compute-Server** (unter LINUX bis zu **8** SWAP Bereiche à 128MB anlegen, mit geringfügigen Modifikationen sogar **64**)
- **Simultane Logins/Festplatten**: **Parallelisierung** (Lasten auf div. HDDs verteilen, SCSI Controller Feature disconnect benutzen, evtl. RAID Controller), **RAM erhöhen** (für dynamische Puffer: read aheads oder delayed writes)
- **Sicherheitsaspekte**: möglichst kleine ROOT-Partition anlegen, um Schreibzugriffe zu minimieren, LINUX zur Integritätssicherung nicht auf DOS Dateisystem installieren (wegen Filesystem Checkprogramme)

**Praktische interaktive Partitionierung**:

1. **Physikalische Bestimmung**: zu bearbeitende Festplatte bestimmen (1. IDE HDD **/dev/hda**, 2. IDE HDD **/dev/hdb**, 1. SCSI HDD **/dev/sda** etc.), Partitionen anlegen und Typen (!)(**LINUX SWAP 82**, **LINUX NATIVE 83**) bestimmen, mit **Festlegen der Dateisysteme** bestätigen
2. **Mountpoints**: (einzelnes Dateisystem, hierarchischer Baumast) und **Devices** festlegen (in **/etc/fstab** ~ LINUX-interne Dateisystem-Tabelle, **msdos**, **vfat**, **UMSDOS** und **HPFS** Partitionen können auch eingegliedert werden), **fstab** kann auch eingelesen werden (z.B. für Update), BOOT Verzeichnisse müssen im ROOT belassen werden (also **/bin**, **/dev**, **/lib**, **/etc**, **/sbin** nicht mounten), Mountpoints müssen mit absoluten Pfaden angegeben werden
3. **Inodedichte**: Speicherplatz für durchschnittliche Dateigröße angeben, viele Inodes für viele kleine Dateien, wenig Inodes für vorwiegend große Dateien, generell ca. **4096bytes** pro Inode (Standard), beste Performance bei Wahl der gleichen Inodedichte bei allen Dateisystemen
4. **Formatierung**: **Normal formatieren** oder **mit Prüfung** bei alten Datenträgern, SCSI Geräte benötigen keine Prüfung

## 4. Softwareausstattung wählen:

**Konfiguration laden**: um eine spezielle Softwarevorauswahl zu treffen, Distributionspakete hinzufügen oder ersetzen (cancelt Vorauswahl)
**Konfiguration ändern/erstellen**: Serienauswahl, einzelne Pakete an- oder abwählen → hier evtl. bereits X-Server aus Serie **xsrv** selektieren
**Installation starten**: YAST installiert Pakete (Softwaregrundsystem)
**Installation abschließen und YAST beenden**: wählen, nachdem Pakete installiert wurden

## 7. Grundkonfiguration:

**Wahl der Kernelversion**: im Zweifel "**Standard (E)IDE Kernel**" wählen
**Zielpfadänderungen**: Kernel default im **/boot** Verzeichnis, Konfigurationstextdatei in **/usr/src/linux**

## 8. Systemstart konfigurieren:

- **Bootdiskette**: unbedingt mit eigener leerer vorformatierter Diskette erstellen, besteht evtl. aus mehreren Kernel (LILO gemanaged) und lädt nur den Kernel, um dann zum HDD Start zu wechseln (im Kernel ist ROOT-Partition als Root-Device konfiguriert, **init**

und **Startskripte** etc. kommen nur von HDD), der Bootdiskettenstart ist unabhängig vom installierten LILO und dem HDD Kernel

- **LILO**: **Linux Loader** (Bootmanager für DUAL-BOOT-Systeme)
  - **DOS, WIN9X**: ohne Probleme, vorausgesetzt für beide OS steht eine primäre Partition unter der **1024-Zylinder-Grenze** zur Verfügung, LILO Installation im **MBR** (Master Boot Record), **/sbin/lilo** stellt einen kaputten MBR wieder her
  - **WINNT**: kein LILO, sondern **NT Bootmanager**: NT Installation → LINUX gewöhnlich installieren und auch einen FAT Datenträger mounten (nicht verfälschte Mountoptionen **conv=auto** oder **conv=text** verwenden) → LILO in **ROOT-Partition** installieren (nicht MBR) → diesen LILO Bootsektor auf FAT-Mountingpoint als Datei übertragen
    /bin/dd if=<rootpartition> bs=512 count=1 of=<FAT-Mountingpoint>\bootsek.lin
    → NT booten und **bootsek.lin** in Hauptverzeichnis NT-Systemlaufwerk kopieren
    → NT **boot.ini** Eintrag: **c:\bootsek.lin="LINUX"**

**LILO**: startet Bootsektoren von Partitionen oder LINUX-Kernel (besonderes Feature)
**LILO-Startmaschinerie**: LILO-Bootsektor (Codeanfang) → LILO-Maschinencode (default in **/boot/boot.b**) → Map-Datei (default **/boot/map**) → Message Datei (optional, in **/boot/message**) → Kernel und Bootsektoren (jeder Schreibzugriff auf Maschinerie killt LILO)
**LILO-Lokationen**: Diskette,**1024**-Zylinder-Grenze beachten (sonst von BIOS-Treibern nicht erreichbar)
**LILO Konfiguration**: Eintrag in **/etc/lilo.conf** (manuell oder YAST-automatisiert, Datei wegen Paßwörtern nur lesbar **chmod 0600** machen)
**Lilo.conf: Aufbau:**

- **Globaler Abschnitt**: allgemeine Einstellungen, **boot=<bootdevice>** (LILO Installationsziel, Flobby/Partition/Platte möglich), **message=/boot/greetings**, **prompt** (erzwingt Prompt), **password=xxxxxxx** (LILO pwd setzen)
- **Systemabschnitte**: Einstellungen pro OS, erstes System in Liste wird automatisch gebootet (nach **delay** oder **timeout**), Systemabschnitte werden durch **image=<kernelimage>** (für LINUX) oder **other** (für andere OS, benötigt **loader=<bootloader>**: Verweis in Map-Datei auf PseudoMBR, dann Start des fremden Bootsektor default **/boot/chain.b**, **table=<ptabelle>**: Quell-Device für Partitionstabelle, die in PseudoMBR soll) eingeleitet, **#** Kommentarzeichen bis Zeilenende, **root=<rootdevice>** (sicherheitshalber, sonst Kerneleintrag), **label=<name>** (15 Zeichen Menüname)

**Weitere Bootmanager**: **System Commander Deluxe**, **Partition Magic**

## Hinweise zum Bootvorgang

**Bootvorgang**: **BIOS** startet (Rumpfsystem überprüft Tastatur, Bildschirm, RAM und **CMOS**) → MBR wird in RAM geladen

**CMOS**: enthält Zeitangaben, Datum und wichtige Peripherie
**MBR**: Master Boot Record, **512 Byte**, erster Datensektor der ersten HDD, Struktur durch OS-übergreifende Konvention festgelegt, erste **446 Byte** für Programmcode reserviert, folgende **64 Byte** Partitionstabelle, **2 Byte** feste magische Zahl **AA55** (sonst ist MBR ungültig)
**Bootsektoren**: jeweils erste Sektoren der HDD Partitionen, ebenfalls **512 Byte**, DOS/WINDOWS/OS/2 nutzt diese, bei LINUX leer (daher ist LINUX Partition nicht von selbst startbar), gleiche magische **2 Byte** Kennung wie MBR
**DOS/WIN9X**: Partitionseintrag im MBR **aktiv** gekennzeichnet (**bootable** → d.h.von hier System laden), DOS-Code im MBR ist erste Stufe des **bootloaders** (**first stage bootloader**: überprüft Partition auf gültigen Bootsektor), Code im Partitions-BR startet als zweite Stufe nach (**secondary stage loader**: lädt OS-spezifische System Programme)

**Weitere Bootkonzepte**: zusätzliche Systeme von Diskette booten, zusätzliche Systeme zur Laufzeit nachladen (z.B. LINUX-Start von DOS mit **loadlin.exe**, NETWARE-Start von DOS mit **server.exe**), Installation eines Bootmanagers

## 9. Sonstige Konfigurationen

Konfiguration von CDROM, Zeitzone wählen (**MET – Middle European Time**), Hardwareuhr festlegen (alternativ zu MET z.B. **GMT Greenwich Mean Time**)
**Netzwerkkonfiguration**: Rechner- (z.B. azrael) und Domainname, (z.B. moonshea.com), Art des Netzwerk (loopback oder echtes Netzwerk: Netzwerktyp, IP-Adresse, Netmask, Gateway, inetd, portmap, NFS Server, From-Zeile für News Posting, Netz-Klient mit Zugriff auf Nameserver (IP, YP-Domain), Kernel-Modul für netzwerkkarte, sendmail.cf für Mailsystem
**Logins und Paßwörter**: root Paßwort
**Modem einrichten**: serielle Schnittstelle angeben
**Maus einrichten**: Typ und serielle Schnittstelle, **gpm** einrichten (Programm zum cut n´paste zwischen Konsolen, kann Probleme mit dem X-Server bereiten)

## Systemkommandos

**Systemneustart**: Kommando **reboot** oder **shutdown –r now** oder **STRG + ALT + ENTF**
**Systemabmeldung**: Kommando **logout**
**Als Benutzer Administratorrechte erlangen**: Kommando **su**
**Als Systremadministrator und Benutzer anmelden**: Wechsel von root, Kommando **login**
**System sofort herunterfahren**: Kommando **system –h now**
**Abfrage des aktuellen Terminals**: Kommando **tty**
**Gesamte Terminalausgabe löschen** : Kommando **clear**

## Anzeige Kommandos

| Kommando | Funktion |
|---|---|
| **whoami** | Zeigt eigenen Loginnamen, evtl. **logname** |
| **who** | Anzeige zu eingeloggten Benutzern |
| **finger** | Anzeige zu eingeloggten Benutzern |
| **date** | Aktuelles Datum / Uhrzeit anzeigen |
| **cal** | Kalender des laufenden Monats anzeigen |

## Terminal Steuerzeichen

| Steuerzeichen | Funktion |
|---|---|
| **Newline**: RETURN | Abschluß / Übernahme Kommandozeile |
| **Eof**: CTRL + D | End of file<br>Abschluß / Übernahme Kommandozeile |
| **Erase**: BACKSPACE, CTRL + H, # | Löscht letztes Zeichen |
| **Kill**: @, CTRL + U | Löscht die gesamte Zeile |
| **Stop**: CTRL + S | Halt der rollenden Bildschirmausgabe |
| **Start**: CTRL + Q | Fortsetzen der Ausgabe |
| **Intr**: DEL, BREAK, CTRL + C | Programmabbruch |

**Stty –a** **Steuerzeichenanzeige**
**Stty erase “%”** **neues Steuerzeichen für erase festlegen**
**Stty erase \^h** **erase auf CTRL + H zurücksetzen**

# 3 Dateisystem

## Festplatten Organisation und Inodes

**UNIX Installation**: benötigt mindestens **2** Partitionen, das **Root File System** und eine **Swap Partition**
**Partitionen**: bestehen aus Datenspeicherungsblöcken, ein **Block** enthält **512 bytes** Daten
**Dateisystem**: **4 Sektionen**: **boot block** (synonym **block 0**, speichert die Boot Prozedur, kann auch leer sein), **super block** (synonym **block 1**, enthält die globale Dateisystemdefinition), **inode list** (synonym **information node list**, **inode**: beschreibt Dateicharakteristiken und den Pointer zum richtigen Datenblock), **data block** (Speicherblock mit Dateiinhalten)
**Inode List**: identifiziert die blocks, die zu einer Datei gehören
**Inode**: Systemstruktur, beschreibt Sicherheit (User und Gruppen IDs), Erstellungs-/Änderungsdatum (Date und Time Stamp), Größe, Speicherort, Berechtigungen einer Datei, in UNIX SVR4 **64 bytes** lang, wenn eine Datei länger als **1 block** ist, enthält die Inode einen **Index** aller betroffenen Datenblockadressen der Datei (Blöcke sind auf der Harddisk verstreut)
**Verzeichnisdatei**: enthält Dateinamen
**Verzeichnistabelle**: um inodes und ihre respektiven Filenamen zu linken
**Adressspeicherung eines Datenblocks**: Inode hat ein **Array** von **13 disk block Adressen** (falls ungenutzt haben diese den **Wert 0**), die ersten **10** Dateiblocks werden getrennt der restlichen Information gespeichert

## Inode Funktionen

Jede Inode hat **13** Pointer, die auf die Datenblockadressen einer Datei zeigen, diese Pointer werden für den Datenblockzugriff benötigt, die ersten **10** Pointer sind direkte Adressen der ersten **10** Datenblocks einer Datei, benötigt eine Datei mehr Blöcke so verweist der **11. Pointer** auf einen weiteren Block der ein **Set** von **256** blocks adressiert (ebenso der **12.** und **13.** Pointer)

**3 gespeicherte Dateizugriffszeiten einer Inode**:

1. letztes Änderungsdatum
2. letzte Ausführung oder Lesezugriff
3. letzter Inodezugriff oder –änderungsdatum

**Dateizugriff über Kernel**: Kernel bezieht sich über den Inode List Index auf die Inodes, er liest die Inode in den Speicher, nach Update wird die Datei zurück ins Dateisystem geschrieben

## Dateiarten

**Datei**: maximale Größe **16GB**, intern betrachtet als **bytestream**, Dateiverwaltung anhand **Inode** (Datenstruktur, enthält administrative Informationen: z.B. Größe, Zugriffsrechte)
**Dateiverzeichnisse**: zuständig für Zuordnung von Dateiname und Inode, hierarchisches Gebilde

**5 Dateiarten**:

- **Einfache (reguläre) Dateien**: **(-)** Text-, Grafik-, Programm- oder Objektcode
- **Dateiverzeichnisse (Directories)**: **(d)** Inhalt entspricht Tabelle mit Dateiname und zugehörige Inode-Nummer
- **Gerätedateien (Spezialdateien)**: **(2 Unterscheidungen: (b)**: **blockorientiertes Gerät (Plattenspeicher, Floppy), (c)**: **zeichenorientiertes Gerät (**synonym **raw device, z.B. Terminal, Drucker, Netzwerkmedium),** externe Geräte: nur über Kommandos editierbar, nicht mit Texteditoren
- **Named Pipes**: **(p)** Röhren, FIFO Dateien, verwendet zur effektiven Interprozeß-Kommunikation
- **Links, Verweise**: **(l)** Mehrfachbenennung, Dateien können über verschiedene Namen angesprochen werden

## Device Special Files

**Features von Character special device files**: Input/Output mit jeweils nur **1** Zeichen gleichzeitig
**Features von Block special device files**: operieren mit logischen blocks á **512**, **1024**, **4096** oder **8192bytes**
→ wenn ein Gerät beide Schnittstellen besitzt, so gibt es auch **2 device files** dafür!

**Major and minor device numbers**: Gerätedateien besitzen **2** einzigartige Nummern zur Unterscheidung, diese sind in **/dev** aufgeführt, **major**: gibt **Geräteklasse** an (alle diese Geräte benutzen den gleichen OS Code), **minor**: **spezielle Adressierung** eines speziellen Gerätes, Parameter um zu lesen oder schreiben

| Standard-Gerätename | Beschreibung |
|---|---|
| **/dev/fd0** | Erstes Diskettenlaufwerk |
| **/dev/hda** | Erste Festplatte am IDE Bus |
| **/dev/hda1** | Partition 1 der ersten IDE HDD |
| **/dev/cdrom** | CDROM Laufwerk als interne Verknüpfung, die durch YAST angelegt wird |
| **/mnt** | Mounting Point (Standardeinbindepunkt) |

## Dateien und Verzeichnisse

**Dateinamen**: **14** signifikante Zeichen, alle Zeichen außer **/** und **Freizeichen** möglich, keine Benennung mit UNIX-Kommandos, Dateinamenserweiterung unnötig aber möglich, case sensitive, Benutzung von Punkten zur optischen Strukturierung, versteckte Dateien starten mit einem Punkt
**Wildcards**: synonym **Jokerzeichen**, **Suchmuster**, **?** (genau ein Zeichen), ***** (beliebig viele Zeichen, kann mehrmals vorkommen), **[...]** (paßt zu genau einem der angegebenen Zeichen an genau dieser Position, kann auch ein Zahlenbereich sein **z.B. [1-9]**), **[!...]** (paßt zu genau einem Zeichen außer den angegebenen)

**Verzeichnisse**:

- **Root**: Wurzelverzeichnis, oberster Punkt des Dateisystems
- **Working Directory**: aktuelles Verzeichnis, aktuelle Arbeitsposition, kann innerhalb Pfadnamen abkürzend mit **1 Punkt** benannt werden (**.**)
- **Home Directory**: Heimatverzeichnis, vom Systemverwalter individuell zugeordnetes Benutzerverzeichnis
- **Login**: erstes Verzeichnis bei login, meistens home
- **Parent Directory**: übergeordnetes Verzeichnis, kann auch mit **2 Punkten** benannt werden (**..**)

**Pfadkennzeichnung**: **absolut** (beginnt immer im root), **relativ** (ausgehend vom aktuellen Verzeichnis)

## Kommandos zur Arbeit mit Dateien

| | |
|---|---|
| **ls** | Verzeichnisanzeige (**-a** zzgl. versteckte Dateien, **-aF** zzgl. Kennung für Verzeichnisnamen und ausführbarer Dateien (**/** oder @), **-l** zeigt lange Dateiinformation, **-i** Anzeige der Inode Nummer, **~** Homeverzeichnis anzeigen, **name** kann an **Tilde** angestellt werden, um andere User-Homeverzeichnisse anzuzeigen), **-R** schließt sämtliche Unterverzeichnisse mit ein, **-s** zeigt Verzeichnisgröße in Blocks |
| **xargs** | Hinter pipe: Erzeugt Liste von Argumenten oder Dateinamen, dann wird ein Befehl mit dieser Liste ausgeführt, -l Kommando für jede Zeile ausführen |
| **cat** | Zeigt ASCII Datei fortlaufend an, **-n** nummeriert die Ausgabe am linken Rand |
| **pr** | Wie cat aber Text wird seitenweise mit Kopfzeile formatiert |

| | |
|---|---|
| **more** | Seitenweise ASCII Dateiausgabe |
| **pg** | Wie more |
| **head** | Gibt ersten **10** Zeilen der ASCII Datei |
| **tail** | Gibt letzten **10** Zeilen der ASCII Datei |
| **less** | Textdateien ansehen, **SPACE** für nächste Seite, ↑↓ um zu scrollen, **q** für quit<br>→ ähnlich wie **more** (more kann aber nicht zurückblättern) |
| **lp**<br>**(LINUX: lpr)** | Dateien drucken (**-d** bestimmter Drucker (**LINUX**: **-p**), Auftrag geht dann in die Druckwarteschlange, **-t** Titel mitdrucken, **-c** gleichzeitig auf HDD ausgeben, **-m** Message wenn Job ausgeführt |
| **pwd** | Zeigt das aktuelle Arbeitsverzeichnis mit Pfad |
| **cd** | **Langform chdir**, Verzeichniswechsel, **cd** „alleine“ bewirkt die Rückkehr ins Home-Verzeichnis, **Fehlermeldungen können nur bei der Langform ausgegeben werden** |
| **mkdir** | **Kurzform md**, erzeugt Verzeichnisse, kann durch „blank“ getrennt mehrere Verzeichnisse auf einmal anlegen, **Fehlermeldungen nur bei Langform** |
| **mv** | Dateien oder Vereichnisse umbenennen und/oder verschieben (Originaldatei erhält einen neuen Namen, **rename**), ACHTUNG: keine Warnung beim Überschreiben, **-b** Sicherheitskopie erstellen |
| **cp** | Dateien kopieren (echte Kopie), **-i** warten auf Bestätigung bevor eine existierende Zieldatei überschrieben wird, **-r** rekursive Kopie (mit Unterverzeichnissen) |
| **rm** | Löscht Dateien (**-i** mit vorheriger Rückfrage, **-r** löscht Verzeichnisse mit Inhalt und Unterverzeichnissen) |
| **rmdir** | **Kurzform rd**, löscht leere Verzeichnisse, **Fehlermeldungen nur bei Langform** |
| **ln** | Verweis / Link erstellen (gleiche links haben gleiche Zugriffsrechte), **-s** legt einen **symbolischen Link** an (synonym **soft link**, zeigt nur auf den Pfad der Quelldatei und funktioniert daher über Dateisystemgrenzen hinweg, zeigt Fehlermeldung wenn Quelldatei gelöscht wird), **Syntax**: **ln originaldatei zieldatei** |
| **mtools** | Kommandos, die DOS oder LINUX Syntax erlauben:<br>**mattrib**, **mcd**, **mcopy**, **mdel**, **mdir**, **mformat** (Low-Level nur über fdformat! Unformatierte Disketten müssen also mit fdformat erstellt werden), **mlabel**, **mmd**, **mrd**, **mren**, **mtype** |

## Datei Linking

Bei mehreren gleichzeitig requesteten Dateizugriffen wird die Datei **geshared** und zu usern **gelinked**

**Filelinks**: aktualisieren nur eine einzige Datei bei Benutzeränderung (Platzersparnis, Versionskontrolle!), links sind **i-numbers** die von Inodes referenziert werden, das **Linkfeld** einer Inode kennt die Totalanzahl der i-numbers

**Dateieinträge in Verzeichnissen**: **Struktur**: sind mit **2** Feldern verknüpft (inode Nummer und Dateiname), auf Dateien kann über gelinkte Dateien mit verschiedenen Pfaden zugegriffen werden (sog. **Hardlinks** oder **reguläre links**)

**Nicht-existierende symbolische Links**: synonym **dangling symbolic link** (dangling = baumelnd) werden mit **rm** gelöscht

## Drucken

**Druckernameparamteter**: nennt spez. Drucker oder Druckerklasse
**3 Druckerklassen**: matrix, daisy, laser
**Bestätigungsoutput**: liefern Sequenznummer
**Kommando lpstat**: Statusabfrage Drucker,
**Syntax**: **lpstat –o [request ID, printer_name, class, user]**,
**Kommando cancel**: Druckauftrag per Request-ID abbrechen, **Syntax**: **cancel request_ID**
**Request-ID**: Kombination aus Druckername und Druckjob

## Typeskript-Datei

Loggt User-Interaktionen mit dem Betriebssystem
**Kommando script**: loggt User-Eingaben und OS-Ausgaben
**Syntax**: **script [–a] file**, **-a** append an existierende Datei

## Dateien splitten

**Kommando split**: Textdateien splitten
**Syntax**: **split [-n] [file [name]]**, Option **-n** gibt Zeilenanzahl pro Datei an (default **1000**), **name**: gemeinsamer Prefix (Suffix), default **x**, Summary in **/temp**,
**Kommando csplit**: Dateien aufgrund ihres Inhalts splitten, liefert neben den Splitdateien eine weitere Datei mit den Sektionen
**Syntax**: **csplit [-fprefix] file arg1 [...argn]**, Argumente des Searchstrings werden in Slashmarks (**'/.../'**) eingeschlossen, Prefix default **xx**, Suffix default **00** – **99** (99: letztes mögliches Maximum), xx00 durchsucht bis zum ersten Auftreten Zeile arg1 und schreibt die Datei exklusiv arg1 etc., Bytezahl der Dateien wird im Output angegeben

## Dateiverschlüsselung

Über das Kommando **crypt**, zur Ver- UND Entschlüsselung, aufgrund Export Regulationen gibt es dieses Kommando nur in den USA, **Syntax**: **crypt < datei_unverschlüsselt > datei-verschlüsselt**, nächste Zeile verlangt Passwort mit Zahlen/Buchstabenkombination

## Zugriffsrechte

→ gelten für Dateien und Verzeichnisse

3 **Benutzerkreise**: **user** (**u**, Besitzer), **group** (**g**, andere Gruppenmitglieder der Besitzergruppe), **others** (**o**, alle anderen)
3 **Zugriffsrechte Dateien**: **read** (**r**, lesen), **write** (**w**, in Datei oder auf Gerät schreiben), **execute** (**x**, Binärdateien oder Shellscripts ausführen)
3 **Zugriffsrechte Verzeichnisse**: **r**: durchstöbern, **w**: Dateien anlegen, **x**: Verzeichnis betreten
→ der Systemadministrator besitzt immer alle Rechte
3 **weitere Dateikennzeichen**: **uid**, **gid**, **sticky**

**umask** — **schränkt initiale (default) Rechtevergabe ein, Erweiterung aber unmöglich, Rechte der Maske muß mit Oktalwerten (r:4, w:2, x:1) kodiert werden**
**Es erfolgt also 666 pro Benutzerkreis minus Maskenwert p. Wert (bei Textdateien), 777 bei ausführbaren Dateien**
umask wirkt nicht rückwirkend, nur für neu erzeugte Dateien
Maskenwert default: **022 → wird also abgezogen → 644: rw---r--**

**chmod** — **vergibt Rechte, besitzt folgende Zuweiskommandos:**

| | |
|---|---|
| **+** | **Recht vergeben** |
| **-** | **Recht entziehen** |
| **,** | **Trennzeichen** |
| **=** | **absoluten Wert zuweisen** |

**chown** — **eigene Dateien an andere Benutzer verschenken, kann nicht selbst wieder zurückgeholt werden**
**Syntax: chown NeuerBenutzer dateiname**
**Option –R hat auch Auswirkung auf Dateien und Verzeichnisse in allen Unterverzeichnissen**

**chgrp** — **Dateien an andere Gruppen zuteilen**

**groups** — **zeigt an, in welchen Gruppen ein Benutzer ist**

## Dateisystem

| | |
|---|---|
| **/bin** | **Programme für den Systemstart** |
| **/sbin** | **Systemprogramme, für den Superuser (root)** |
| **/sbin/init.d** | **Bootskriptverzeichnis zur Anpassung des Systemstarts** |
| **/dev** | **Gerätedateien (/dev/tty: aktueller Bildschirm, /dev/null: Papierkorb-Datei)** |
| **/etc** | **systemspezifische Konfigurationsdateien** |
| **/usr** | **Dateien zur Nutzung von anderen Systemen über das Netzwerk (/bin, /sbin, /include, /lib, /share, /ucb)**, in **/bin** und **/sbin** sind die meisten UNIX Kommandos abgespeichert, für jeden Benutzer zugänglich |
| **/home** | **Homeverzeichnisse der Benutzer** |
| **/tmp** | **Zwischenablage** |
| **/var** | **veränderliche Systemdateien (/adm, /spool, /mail, z.B. Warteschlange)** |
| **/stand** | **unter LINUX: /boot: Boot-Dateisystem, enthält Kernel** |
| **/opt** | **wahlfrei einrichtbare Programme** |

→ das Dateisystem ist physisch über mehrere Datenträger/Netzwerk verteilt
**Mounten**: Erreichbarmachung von physischen Laufwerken, das Root-Dateisystem des neuen Laufwerks wird eingehängt, die ursprüngliche Wurzelposition wird verdeckt, Mounten benötigt Administratorrechte

| | |
|---|---|
| **mount** | **zeigt aktuelle Mountpunkte, Optionen: -r nur Leserechte, -t Dateisystem für Mountpoint festlegen** |
| **df** | **zeigt Speicherbelegung der Dateisysteme** |

**Diskettenlaufwerk mounten**:
1. **Verzeichnis anlegen**: **mkdir /floppy**
2. **Mounten**: **mount /dev/fd0 /floppy** (ohne Gerätename ist **/floppy** auch default!)
3. **Unmounten**: **umount /floppy** (Gerätename oder Einbindepunkt angeben, Gerät darf nicht in Gebrauch sein)

→ mit den **mtools** kann über DOS-Modus auch ohne Anbindung über **mdir a:** auf das Floppy zugegriffen werden

**Achtung**:
→ CDROM oder Floppy müssen vor dem Medienwechsel ausgehängt werden. Bei Disketten ist darauf zu achten, daß Daten erst beim Unmounten tatsächlich auf Disk geschrieben werden.
→ Die letzten **10%** jeder Platte sind für den Systemadministrator „root“ reserviert, damit er noch das System in einen arbeitsfähigen Zustand bringen kann, wenn wichtige Platten voll sind
→ Geräte können nicht mehrmals gleichzeitig eingebunden werden
→ CDROMs können nur gemountet werden, wenn keine Audio-CD eingelegt ist

**2 Möglichkeiten zum automatisierten Mounten bei Systemstart**:
- Über das Bootskript **/sbin/init.d/boot.local** (ähnlich wie **autoexec.bat** unter DOS), Kommandos an Dateiende anfügen
- Geräteeintrag in **/etc/fstab** anfügen, jede Zeile setzt sich aus **Gerätename**, **Einbindeverzeichnis**, **Dateisystem** und weiteren **Optionen** zusammen

**Manuell spezifizierte Dateisysteme beim Mounten**: **mount -t dateisystem**, **affs** (Amiga Fast File System), **ext2** (Second Extended Fileystem, LINUX default), **hpfs** (OS/2), **Iso9660** (default CDROM), **minix** (Dateisystem für LINUX Disketten), **msdos** (DOS), **ncpfs** (Novell Laufwerke), **nfs** (Network File System), **proc** (virtuelles Prozeßdateisystem), **smbfs** (Server Message Block, NT, Lanmanager), **sysv** (SCO UNIX, XENIX, andere kommerzielle), **ufs** (BSD, SunOS, NeXTstep), **umsdos** (UNIX on MSDOS, FAT-Aufsatz mit UNIX Funktionalität), **vfat** (FAT32, wichtig für lange Dateinamen)

**Diskettenformatierung**: DOS/Windows Formatierung oder LINUX Form. Möglich, **fdmormat –n /dev/fd0**, ohne **-n** erfolgt eine anschließende Überprüfung, bei Fehlern oder Problemen **fdformat /dev/fd0h1440** (**3,5"** disk, **1,44MB** Speicherkapazität)

## Nützliche Kommandos und Tools

| | |
|---|---|
| **find** | **Dateien suchen, Optionen:** |
| | **/ ab wo** |
| | **-name xxxx Dateiname** |
| | **-user xxxx Besitzer** |
| | **-size blockanzahl** |
| | **-type Dateiart** |
| | **-print Ausgabe auf dem Terminal** |
| **file** | **Art des Dateiinhalts** |
| **cmp** | **byteweiser Vergleich zweier Dateien** |
| | **andere Vergleichskommandos: diff, comm** |
| **locate** | **herausfinden, in welchem Verzeichnis eine spezifizierte Datei sich befindet, dieses Programm sucht in einer eigens erstellten Datenbank, Aktualisierung der Datenbank als root mit updatedb möglich** |

## Dateivergleiche

### ASCII Dateien

**comm Kommando**: vergleicht **2 sortierte Dateien** (gleiche Felder, gleiche Ordnung), zeigt **einzigartige** und **identische** Zeilen beider Dateien an (zeigt beide Zeilennummern an!), Ausgabe in **3** Spalten (in der **3. Spalte** sind die gemeinsamen Einträge, in den ersten beiden die unterschiedlichen), **Syntax**: **comm [option] datei1 datei2**, Option **-1**, **-2** oder **-3** unterdrückt die Bildschirmausgabe der betroffenen Spalte, **Syntax wenn als Filter benutzt**: **comm - filename** (**-**: Standardinput: Eingabe in nächster Zeile, **CTRL + D** zum terminieren)

**diff Kommando**: **generiert einen Bericht** bei Vergleich, effizienter Weg um Dateien identisch zu machen, **Syntax**: **diff [options] datei1 datei2**, **Optionen**: **-f** produziert ein Skript in entgegengesetzter Richtung (ausgegebene Nummern sind Zeilennummern, Buchstaben: **a append** in datei1 um anzugleichen, **c change**, **d delete** in datei1 um anzugleichen), **-b** ignoriert angehängte Freizeichen (**<** zeigt Inhalte aus datei1, **>** aus datei2), **-e** produziert Skript mit **ed** Kommandos um datei2 aus datei1 wiederherzustellen, **-r** rekursiv für Dateien in gemeinsamen Unterverzeichnissen, **Syntax**: **diff -r dir1 dir2**,

**bdiff Kommando**: um **sehr große Dateien** (**> 3500 Zeilen**) zu vergleichen, zerlegt Dateien in Segmente, startet den Vergleich in der ersten ungleichen Zeile, **Syntax**: **bdiff datei1 datei2 [n]**, **n** ist der Zeilenzerlegungsfaktor (default **3500**), dateix kann auch durch die Standardeingabe ersetzt werden (**-**)

**diff3 Kommando**: **drei Dateien vergleichen**, **Syntax**: **diff3 [options] dat1 dat2 dat3**,
**====** alle **3** Dateien sind unterschiedlich
**====1** Datei1 ist ungleich zu den anderen beiden
**-e** Option schreibt alle Unterschiede aus Datei2 und Datei3 in Datei1

**sdiff Kommando**: **sdiff [-w n] dat1 dat2**, **-w** Weite der Ausgabezeile (default **130 Zeichen**), **-s** undrückt die Ausgabe gleicher Zeilen, **<** nur in dat1 vorhanden, **>** nur in dat2, **|** unterschiedlich

### Dateien mit ASCII-Fremdzeichen

**cmp Kommando**: **cmp [options] dat1 dat2**, zeigt nur die Stelle des ersten Unterschieds, Zeilennummer (#) und Zeichennummer (@) werden angezeigt, **-l** zeigt Bytenummer in dezimal

und unterschiedliche bytes in oktal, **-s** silent ohne report, Achtung: auch leere Zeilen sind ein Unterschied
Returncode für identisch: **0**
Returncode für unterschiedlich: **1**

**od Kommando**: **octal dump**, nicht-ASCII Dateien im ASCII Format anzeigen, **od [-c] [filename] [offset]**, **-c** im ASCII Format anzeigen, **-b** jedes byte oktal ausgeben, **-d** gibt **16bit** Wörter als unsigned decimal aus, **-o 16bit** als unsigned octal (default), **-x** Ausgabe als Hexadezimal, **offset**: wo der dump starten soll **[+]number[.][b]** (als Block (**b**) oder Bytenummer (ohne b), oktal (ohne .) od. dezimal (.), **+** falls kein Dateiname angegeben)

## Duplikater Inhalt in Textdateien

**Kommando uniq**: vergleicht benachbarte Zeilen (zuerst also sortieren!), **Syntax**: **uniq [options] filename**, **Optionen: -u** zeigt nur einzigartige Zeilen an, **-d** zeigt nur Duplikate, **-c** zeigt alle Zeilen mit der Vorhandenheitszahl der Zeilen, ohne angegebenem Dateiname wird der Standardinput benutzt, ohne Angabe von Optionen werden die Zeilen in die Outputdatei kopiert

## Selektive Extraktion von Dateien

**Kommando cut**: Filter, um spezielle Spalten oder Felder einer Datei zu extrahieren
**Felddefinition**: Zeichenstring mit **Fielddelimiter** (default TAB), **Syntax**:
**cut [options] filename**, **Options**: **-c*Zahl*** extrahiert ein bestimmtes Zeichen in Folge, **-f*1,2*** gibt erstes und zweites Feld aus (**-f4-** alle Felder ab dem vierten), **-d","** Komma ist Fielddelimiter

**Kommando paste**: Zeilen (als Tabellenspalten betrachtet) aus verschiedenen Dateien verketten,
**Syntax**: **paste [-d" "] datei1 datei2**, ein Bindestrich als Dateiname ersetzt eine Datei und fordert Eingabe vom Standardinput

## Speicherplatz

| | |
|---|---|
| **df** | **zeigt proz. Speicherbelegung in einem Dateisystem, Optionen wie Angabe eines Gerätes oder Mountpoint möglich, -t total allocated block figures, akzeptiert auch Datei- und Verzeichnisnamen** |
| **du** | **zeigt Speicherbelegung einer Datei oder Verzeichnisbaums, ohne Parameter bezogen auf das aktuelle Verzeichnis, berichtet in 512byteblocks**<br>**Option**: **-a** jede Datei (ansonsten nur Subdirs), **-s** total disk space |
| **free** | **zeigt Arbeitsspeicherinformationen** |
| **top** | **zeigt die gesamte Systemauslastung inkl. Prozesse, aktualisiert im 5sec Takt** |

## Archivierung

| | |
|---|---|
| **tar** | syn. **Tape Archiver, unkomprimierte Archivierung**<br>**Optionen: c: create - erzeugen, r: Anfügen einer Datei ans Archivende, x: extract - Extrahieren, t: table - Inhalts-Anzeige, f: file - Spezifikation eines Dateinamen (**muß immer die letzte Option sein**), v: verbose - zeigt während dem Einpacken alle Dateien an**<br>(sofern default nicht erwünscht) |
| **cpio** | **Datei- oder Verzeichnisgruppen kopieren, Optionen: -o kopieren und erstellen einer Archivdatei mit zusätzl. Informationen, -i lesen, -p kopieren, -v Anzeige der Dateinamen, -d Herstellung der ursprünglichen Verzeichnisstruktur** |

## Kompressoren / Packer

**Compress/decompress/zcat** (Anzeige): Option **–v**: zeigt eingesparten HDD Platz Report an, **.Z** Erweiterung, reduziert **50-60%** schnell

**Pack/unpack/pcat** (Anzeige): **.z** Erweiterung, reduziert **60-75%**, pcat kann auch mehrere Dateien gleichzeitig behandeln
**Gzip/gunzip/zcat** (Anzeige)

# 4 Editoren unter UNIX

**2 Unterscheidungen**:
- **Zeilenorientierte Editoren**
- **Bildschirmorientierte Editoren**: benutzen Cursor zur Markierung

**Editoren**: **ed** (zeilenorientierter Standardeditor), **ex** (Erweiterung von ed), **vi** (bildschirmorientiert), **emacs**, **TeX/LaTeX**

## vi Editor

**2 Modi**:
- **Kommandomodus** (default)
- **Eingabemodus** (in Kommandomodus mittels **ESC** zurückswitchen)

**Dateibearbeitung**: immer zuerst Bearbeitung im Puffer, benötigt Befehl (Speicherung) um Modifikationen direkt in Datei zu schreiben, vi verzichtet auf spezielle Funktionstasten (z.B. **F1**)
**Einstellungen**: in **.exrc** im Login Verzeichnis, Einstellungsanzeige in vi mit **set all**

## vi Kommandos

Kommandos werden nicht als Echo ausgegeben, das Voranstellen von Zahlenwerten an Befehle erlaubt mehrfache Wiederholungen
**Optionen**: **a oder i**: Text einfügen, **x**: löschen, **D**: Zeile löschen, **ZZ**: speichern und beenden, **:q!**: quit, **:w**: speichern und vi nicht verlassen
**Zwischenspeicher**: bleibt auch bei Dateiwechsel erhalten

# 5 Prozeßsystem

## Jobs

**Job**: Aufgabe, die der Computer bewältigt, in der **Bourne Shell** wird die Jobausführung durch die **Job-Shell** oder **jsh** kontrolliert, die **C-Shell** übernimmt diese Aufgabe allein, die **Korn-Shell** bietet Jobverwaltung optional an
**Job Control**: erlaubt zwischen Vorder- und Hintergrundjobs umzuschalten
**Jobs aussetzen**: **CTRL + Z** (**suspend**), um einen eiligeren Job auszuführen
**Einen suspendierten Job wiederaufnehmen**: Kommando **fg** (**resume**)
**Jobs auflisten**: Kommando **jobs** zeigt initiierte und gestoppte Jobs an (**+** Zeichen zeigt den derzeitig laufenden Hintergrund-Job an, der nächste erhält ein **–** Zeichen)
**In UNIX SVR4**: höhe Prioritäten werden durch **–** Zeichen (diese Jobs erhalten schnelleren Zugriff auf kritische Ressourcen), niedrige durch **+** Zeichen ausgedrückt, die **niedrigste Priorität ist 19**, **die höchste –20**, diese Werte können je nach **nice** Version variieren, Syntax des **nice** Kommando um Priorität zu verändern:
**nice [-|+]n command**

## Prozesse

→ Ausführbarer Programmcode kann zweimal gleichzeitig als unabhängige Prozesse ausgeführt werden

**Prozeß**: was im Rechner läuft, eine in Ausführung befindliche Instanz, eine Folge von Zustandsänderungen, die bei Abarbeitung eines Programms durch einen Prozessor zustandekommt, Prozesse haben Auslöser!

**Parallele Prozesse**: nicht unbedingt wirklich parallel sondern **zeitlich verschachtelt**, Entscheidung welcher Prozeß in einem Moment bearbeitet wird heißt **scheduling**, häufig realisiert durch Prioritätenverteilung, unter UNIX heißt das Verfahren:
**Time-Sharing-Verfahren**: CPU wird für eine Zeitscheibe (kurzes Zeitintervall) an einen Prozeß vergeben, wenn Prozeß seinen Job während der Dauer der Zeitscheibe nicht abarbeitet, muß er sich neu bewerben. Dieser Wechsel heißt **Prozeßumschaltung** (**task switch**) und muß ebenfalls effektiv implementiert sein (da dieser auch Zeit benötigt). Bei starker Prozeßverlangsamung spricht man von **Last**

## Prozeßverwaltung

Prozeßinformationen (z.B. Name oder Identifikationsnummer, Priorität, zugehöriger Code, Speicherstartadresse) werden in internen Datenstrukturen (sog. **Prozeß-Kontrollblöcken PDB**) gespeichert.

**3 Unterscheidungen von Grundzuständen**:
- **Aktiv** (running)
- **Bereit** (ready): wartet auf CPU Zuteilung
- **Wartend** (waiting): auf Ereignis/Bedingung

Prozesse befinden sich in einer **Warteschlange**.

## UNIX Prozeßsystem

Prozesse erhalten eine eigene Nummer: den **PID** (**Prozeß-Identifikator**). Prozesse sind hierarchisch (der oberste Prozeß ist der **Kernel-Prozeß** mit der **PID 0**, dieser startet den **Init-Prozeß** mit **PID 1** usw.). Jedes angeschlossene Terminal erhält einen **Sohn-Prozeß** (sog. **getty-Prozeß**): **Parent-Child-Prinzip**

**Prioritäten**: werden in gewissen Abständen durch Ausgleichsalgorithmen neu berechnet (abhängig von CPU Zeit, Wartezeit und Anfangspriorität), **Dialogprozesse** (mit E/A Operationen) werden in der Regel angehoben, **rechenintensive Prozesse** ohne E/A werden herabgesetzt

## Prozeßkommandos

**Kommando ps**: zeigt Prozesse an, **Optionen**: **-l**: Langform, **-f**: volle Form, **-p**: Info für Prozesse mit nachfolgender Nummer, **-t**: Prozesse eines bestimmten Terminals, **-u**: Prozesse eines Benutzers, **-efl**: alle Prozesse werden ausgegeben (siehe Seite 53, ebenfalls der Schalter **-a**)

**Kommando pstree**: Prozeßliste als hierarchische Struktur anzeigen

**Hintergrundprozesse**: läßt man laufen durch Anfügen von „ **&**“ an Kommandos, hier empfehlen sich **E/A Umlenkungen** (**<** oder **>**)

**Kommandos zur Signalübergabe an Prozesse**: Signalversendungskommando **kill** (kann nur auf eigene Prozesse angewendet werden, **z.B. kill –9 470**), **Optionen**:
**2 SIGINT** (del Standardbehandlung), **9 SIGKILL** (unbedingtes Beenden, kann nicht ignoriert werden, entlädt in einem Rutsch die primäre bash shell und alle nachfolgend geladenen Programme und Anwendungen, auch X Window, nicht gesicherte Daten gehen verloren), **15 SIGTERM** (Beenden)

**Prozeßpriorität vergeben**: root darf relativ erhöhen, user können nur herabsetzen, Kommando **nice** zur schrittweisen (**langsam**: keine Optionen angeben) oder sofortigen Änderung der Basispriorität (Differenzwert zwischen **0** und **20**)

**Systemspezifische Prozesse**: sind meistens im Zustand **sleeping** (wartend) und werden bei Bedarf für die Bedarfsdauer aktiviert, diese zyklischen Systemprozesse nennt man **Dämonen** (engl. **Daemon: disk and execution monitor**), haben in der Prozeßübersicht meist ein **Fragezeichen/Bindestrich** in der **TTY Spalte**, z.B. init-, getty-Prozeß oder Print Spooler, swapping/paging

### Automatisierung der Jobausführungszeit

Kommando **at**, Syntax:
**at time [month month day] [day of week] [week] [file]** → danach springt der Cursor in die nächste Zeile und wartet auf Kommandoeingabe, Eingabe abschließen mit **CTRL + d** (case sensitive)
**Argumente für time**: **am**, **pm**, **n** für noon und **m** für midnight, ist die Zeit nicht mit Argumenten spezifiziert, so wird die 24-Stunden Regelung benutzt, ohne spezifizierten Tag wird die nächstmögliche Ausführungsstundenzeit gewählt
**Beispiel**: **at 11am Jun 30**

# 6 Shell

die Shell ist der UNIX Kommandointerpreter (ein eigenes Programm), GUIs bauen auf Shell Funktionen auf, in verschiedenen Fenstern können mehrere Shells geöffnet werden

**Shell-Arten**:
- **Bourne-Shell**: erste UNIX Shell, Programmname **/bin/sh**, **$** Promptzeichen
- **C-Shell**: BSD Shell, Programmname **/bin/csh**, **%** Promptzeichen
- **Korn-Shell**: weiterentwickelte Bourne-Shell, Programmname **/bin/ksh**, **$** Promptzeichen
- **Bourne-Again-Shell**: LINUX Standard, Programmname **/bin/bash**, Promptzeichen **$** od. **>**
- Div. Varianten **restricted Shells**

**Abarbeitung einer Kommandozeile**: Kommandozeilenzerlegung, Parameterersetzung und Variablenauswertung, Kommandosubstitution, Trennzeichenzerlegung, Auswertung von E/A Umlenkungen, Dateinamen-Expandierung
**Nichtgefundene Befehle**: werden als externes Programm gesucht in **/bin** bzw. **/usr/bin**

### E/A Umlenkung

Programme sind bei ihrer Ausführung mit **3** E/A Kanälen verbunden, die standardmäßig einem Terminal zugeordnet sind. Diese Zuordnung kann über E/A Umlenkung geändert werden
- **stdin**: **Standardeingabe** (Dateideskriptor **0**) ~ alle Eingaben
  - Umlenkung: **kommando <datei**
- **stdout**: **Standardausgabe** (Dateideskriptor **1**) ~ alle „normalen“ Ausgaben
  - Umlenkung: **kommando >datei** bzw. **>>datei**
- **stderr**: **Standardfehlerausgabe** (Dateideskriptor **2**) ~ alle Fehlermeldungen
  - Umlenkung: **kommando 2>datei**

**E/A Umlenkung**: Abänderung der normalen Tastatureingabe bzw. Bildschirmausgabe z.B. in Dateiausgabe (falls Name vorhanden, wird diese ohne Warnung überschrieben)

| | |
|---|---|
| **>** | Datei-Neuerzeugung, Überschreibung |
| **>>** | Ausgabe an Datei anhängen |
| **2>** | Umleitung Fehlerausgabe (bei Bourne und Korn Shell) |
| **>&** | Umleitung Standard- UND Fehlerausgabe |

**Bei Kombinationen von E/A Umlenkungen**: erst Input-, dann Output-Redirection abhandeln

**Noclobber**: um Dateiüberschreibungen zu vermeiden, in der **C Shell** ist es eine **Variable**, in der **Korn Shell** eine **Option**, UNIX verneint die Aktion bei gesetztem **noclobber** Wert

- **C-Shell**: **set noclobber**, Abhilfe durch **Ausrufezeichen**: **command file1 >! File 2**, Ausrufezeichen überschreibt den noclobber Effekt
- **Korn-Shell**: **set –o noclobber**, Abhilfe schaffen durch **Piping**: **command file1 >| file2**, Piping übsrschreibt den Effekt der noclobber Option

## Kommandoverknüpfung zu Pipes

Mechanismus, bei dem Ausgaben eines Kommandos/Programms als Eingabe für ein zweites dienen, Pipezeichen **|**, Realisierung über Puffer, der Nachteil entfällt, eine mediäre Datei wie bei der E/A Umlenkung zu erstellen

**Zwischenresultate in Dateien auf HDD schreiben**: über Kommando **tee filename**
(Mechanismus funktioniert über Verdopplung der **stdin** (Standardeingabe))
**Filter**: werden verwendet zwischen 2 Pipes

## Bourne Shell

**Build-in Shell Kommandos ohne Prozeßerzeugung**: **echo**, **pwd**, **set**, **export**, **cd**
**Shell Variablen**: zur Abspeicherung von Zeichenfolgen (**strings**), es gibt vordefinierte und benutzerdefinierte Variablen, Variablenzugriff der Shell erfolgt durch **$** in der Kommandozeile, die Shell ersetzt die Variable durch den aktuellen konkreten string

| | |
|---|---|
| **$** | PID der aktuellen Shell, *nicht überschreibbar* |
| **!** | PID des letzten Hintergrundprozeß, *nicht überschreibbar* |
| **-** | aktuelle Optionen der Shell, *nicht überschreibbar* |
| **TERM** | Terminaltyp |
| **HOME** | Home Verzeichnis |
| **PATH** | Suchpfade der Shell |
| **PS1** | Prompt der Shell |
| **PS2** | Fortsetzungsprompt |
| **IFS** | Trennzeichen für Worte in der Kommandozeile |

**Gültigkeit von Variablen**: lokal innerhalb der Shell, Globalisierung durch **export**, **set** zeigt lokal definierte Variablen an, **env** listet alle Variablen des Umgebungsbereiches
**Argumentvariablen**: ihnen können keine Werte zugewiesen werden, sondern erhalten diese indirekt über Parameter-Ersetzung in Reihe der Benutzung (Namen **1** bis **9**), **#** Anzahl der aktuell belegten Argumentvariablen, ***** alle aktuellen Argumente als eine Zeichenkette
**Kommandosubstitution**: Einfügen von Kommandoausgaben in Kommandos über **back-quotes** **`kommando`**
**Quoting**: Entwertungsmechanismen, lexikalische Schutzzeichen, um Sonderwirkung vordefinierter Zeichen auszuschalten, backslash **** (entwertet das unmittelbar folgende Zeichen, vor **Newline** für längere Kommandos), einfache **'** (ersetzt alle dazwischenliegenden Sonderzeichen) und doppelte **"** Hochkommas (wie einfaches Hochkomma aber ohne Entwertung von \ ´ und $)

## Shell Scripts

Enthalten nur ASCII Text, Aufruf eines Shellskripts in einer Subshell mit **sh skriptname parameter** (benötigt x Recht) oder ohne Subshell sofern die PATH Variable ein **leeres Feld** oder ein **Punkt** enthält oder erzwungen in der aktuellen shell mit **. skriptname**

**Formulierung von Bedingungen**: das Kommando **test**, Syntax:
**test bedingung** oder **test [ bedingung ]**
**Operatoren**: **-a** AND, **-o** OR, **!** NOT, Vorrangregelungen über runde Klammern (diese müssen natürlich bei Gebrauch entwertet werden)
**Optionen**: **-r** Leserecht etc., **-f** reguläre Datei, **!=** ungleich, **-eq** äquivalent, **-gt** größer, **-lt** kleiner,

**-ge** größer gleich, **-le** kleiner gleich
**Rückkehrcode**: **$?**, wenn fehlerfrei **0** sonst **1**
**Arithmetische Ausdrücke**: Zahlenvergleiche mittels **test**, Berechnung mittels **expr** (Leerzeichen beachten, Metazeichen müssen entwertet werden)
**E/A Ein-/Ausgabe innerhalb von Shell Skripts**: Kommando **read**, Zuordnung der Eingaben zu Variablen erfolgt wortweise (IFS Trennzeichen) bzw. die letzte Variable erhält den Rest, **line** liest eine ganze Zeile ein, **echo** sorgt für die Ausgabe

**Ablaufsteuerung**: Schlüsselwörter benötigen eigene Zeilen oder müssen durch ein vorstehendes Semikolon getrennt werden

- **Verzweigung**: **if [ bedingung ] ;then kommando(s) ;else kommando(s) ;fi**
- **Zyklen / Schleifen**:
    - **Zählschleife**: **for Zählvariable in Liste ;do kommando(s) ;done** (in-Liste enthält alle annehmfähige Werte als Aufzählung oder in einer Variablen, sonst Standardbehandlung)
    - **Bedingungsschleife**: while [ bedingung ] ;do kommando(s) ;done

| | |
|---|---|
| **break** | Schleife unverzüglich abbrechen |
| **continue** | fortfahren |
| **exit** | Shell Skript beenden, eine hinter exit stehender Parameter (Zahl) wird als Rückkehrcode interpretiert, fehlt er so ist es der RC des letzten Befehls |

### Anlaufdatei .profile

→ privates anwenderspezifisches Loginskript im home Verzeichnis, das nach dem allgemeinen systemweiten /etc/profile gestartet wird

## 7 (Text-)Filter Kommandos

**Wörter eines Textes zählen**: **wc** (Ausgabe: Zeilen, **-w**: Wörter, Zeichen)
**Text sortieren**: **sort**, sortiert zeilenweise, betrifft nur den AUSGABETEXT, nicht die EINGABE! Optionen: **-o name** Ausgabedatei angeben, **-A** ASCII Sortierung und nicht nach Spracheinstellung, **-f** Klein/Großbuchstaben wandeln, **-r** Sortierreihenfolge umkehren, **-i** nicht druckbare Zeichen ignorieren, **-n** Sortierung numerischer Felder nach arithmetischen Wert, **-t TZ** TZ Trennzeichen verwenden, **+1** ein Feld überspringen, **-k3,3n** nur anhand 3. Feld sortieren (wie **+2 –3**)
**Zeichen löschen oder ersetzen**: **tr**, z.B. **tr "abc{}“ "ABX()“** oder **tr '[:upper]' '[:lower]' <datei**
**Zeichen aus einem Text herauskopieren**: **cut**, Optionen: **-c spaltennummer**, **-f feldnummer**, **-d trennzeichen**

### Reguläre Ausdrücke

→ suchbare Pattern in einer Datei, werden in Doppelanführungszeichen eingeschlossen "xxx"
**Besondere Verwendungzeichen**: **eckige Klamern []**, **caret sign ^** (außerhalb eckiger Klammern sucht nach einem Pattern in jedem Zeilenbeginn, in eckigen Klammern eingeschlossen bedeutet daß ein Pattern ausgeschlossen ist), einem Pattern **nachgestelltes Dollarzeichen $** (spezifiziert erforderliches Pattern am Zeilenende), einem Pattern **nachgestellter Punkt .** (dem Suchmuster kann irgendein genau ein Zeichen folgen), **vorangestellter Backslash ** (unterdrückt besondere Eigenschaften vordefinierter Zeichen, Neutralisierung)

**1 Textmuster suchen**: Filterung durch **grep**, synonym **global searching for regular expressions and printing**, **Optionen**: **-i** ignoriert Groß/Kleinschreibung, **-n** liefert alle passenden Zeilen mit der Zeilennummer aus, **-c** liefert nur die Gesamtzeilenzahl des Matchings, **-v** druckt

alle Zeilen die nicht auf das Muster passen, **Syntax**: **grep [options] regular expressions [filename(s)]**,

**Mehrere Textmuster suchen**: **fgrep**, synonym **fast grep**, **Syntax**: **fgrep [-f filename] "[string1] [string2] [...]"**,

**Nach zwei unterschiedlichen strings suchen**: **egrep**, synonym **extended grep**, **Syntax**: **egrep pattern filename**, das **Pipesymbol** separiert unterschiedliche Subexpressions im search string, grouping über **parenthesis** (runde Klammern, **z.B. (ne)***), **angehängtes Pluszeichen** spezifiziert Zeilen mit einem oder mehreren gleichen Zeichen (**z.B. +Z**), **angehängtes Fragezeichen** matcht Zero oder nur das vorgestellte Zeichen, **Hyphen** kann nicht zur Zeichenspektrumangabe benutzt werden

# BSI2

# VERNETZTE UNIX SYSTEME

**Datenübertragungsverbindung**: Standleitung (dauerhaft), Leitungsvermittlung (temporär)
**Optimierte Auslastung**: Paketvermittlung

## Verteilte Dateisysteme

- **NFS**: **Network File Sytem**, kann als normales Dateisystem aufgesetzt werden, von SUN Microsystems entwickelt, um unterschiedlichen Architekturen mit unterschiedlichen OS (MS DOS bis VMS) den resource exchange zu ermöglichen, Resourcen auf Dateiebene, Verzeichnisbaumebene oder Dateihierarchieebene, auch special device files (aber keine Peripherie!), Mountpunkte sind nicht auf shares beschränkt
- **RFS**: **Remote File System**, entwickelt von AT&T, um Resourcen (Ordner + Geräte) freizugeben, Pfadnamen werden identifiziert (Ordner erhalten einen **Resource Identifier**: Netzwerkfreigabename), **share** Kommando, nach einem erfolgreichen Security Check können diese shares lokal gemountet werden
- **VFS**: **Virtual File System**, neueste Entwicklung, befähigt den Kernel zum **Dateisystem-Mapping**, baut auf sog. **streams** (E/A unterstützende Dateien, besitzen Gerätetreiber für Laufzeitkonfiguration) auf, I/O Operationen und Speichermanagement werden verbessert, benutzt außerdem symbolische Links

## UNIX Netzdienste

| Netzdienste | Aufgaben |
|---|---|
| **R-Kommandos**<br>**TELNET**<br>**UUCP** | Anmelden an einem entfernten Rechner und Durchführung einer UNIX Sitzung |
| **R-Kommandos**<br>**UUCP**<br>**FTP** | Dateiübertragung |
| **E-Mail** | Elektronische Post |
| **USENET-Netnews** | Artikel in Newsgroups |
| **ARCHIE**<br>**GOPHER**<br>**WAIS**<br>**WHOIS**<br>**WWW** | Informationssuche |

## R-Kommandos

**Remote Commands**, eigener TCP/IP Anwendungsdienst, haben den Buchstaben "**r**" vor Programmnamen, die Kennung eines Benutzers muß über Rechnergrenzen hinaus gleichgesetzt werden (Sicherheits-riskante Äquivalenzerklärung von Rechnernamen durch den Administrator in **/etc/hosts.equiv**, **1. Auswertung**), ebenso können Benutzer die eigene Kennung entfernter Rechner als äquivalent erklären sofern er dort ein Login besitzt (Eintrag in **$HOME/.rhosts**, **2. Auswertung**)
**R-Anmeldung**: über **rlogin hostname**, Beendigung der Verbindung mit **exit** oder **CTRL-D** oder evtl. der **Escape Zeichenfolge** "**~.**"
**Starten einer Remote-Shell**: wenn nur ein einzelnes Kommando ausgeführt werden soll, Kommando **rsh** oder **remsh** (sofern sich rsh remote auf eine **restricted shell** bezieht), wenn Kommandos mit Sonderzeichen entfernt ausgewertet werden sollen, müssen diese entwertet werden
**Entferntes Kopieren**: **rcp**, ermöglicht den Austausch von Dateien lokal/remote, Syntax:
**rcp datei hostname:$PATH**,
**Entfernte Benutzer lokaler Maschinen auflisten**: **rwho** listet die Namen über den **rwhod Dämon-Prozeß**, der in gewissen Zeitabständen Informationen abfragt
**Entfernte Rechner auflisten**: Onlinezeit aktiver Rechner und Auslastung ermitteln über **ruptime**

## Terminalemulation (Telnet)

→ interessant, wenn der entfernte Host über mehr Eigenschaften als der lokale verfügt
**Prinzip**: Client – Server, **Telnet Client – Telnet Dämon**, Zeichen werden vom Terminalprozeß über TCP an den Server weitergereicht, so können Anwendungen und Kommandos des entfernten Systems benutzt werden, es gibt keine Gleichsetzung der Benutzerkennungen wie bei den R-Kommandos
**Telnet-Kommandos**: **open** Verbindung herstellen, **close** Verbindung beenden, **display** Einstellungen und Steuerzeichen anzeigen, **set** Steuerzeichen ändern, **toggle** Einstellungen ändern, **status** zeigt Übertragungsstatus, **mode** Übertragungsmodus einstellen, **?** Hilfe, **send** Telnet-Sequenzen senden, **quit** Telnet beenden

**2 Übertragungsmodi(und mode Optionen)**:
- **Character-at-a-time** (Einzelzeichen-basiert)
- **Line-by-line** (Zeilenweise)

→ Übertragungsmodi können mit **status** abgefragt werden, **mode line** wechselt zum Zeilenmodus
**Steuerzeichenbehandlung**: Einstellung **localchars** (wenn default ausgeschaltet werden Steuerzeichen von lokal auf remote übernommen, falls eingeschaltet wandelt telnet remote die Sequenzen)
**Verschlüsselte Telnet Alternative**: **ssh**

## Filetransfer (FTP)

→ Dateiübertragung
**FTP Kommandos**: **open**, **quit** od. **bye** od. **CTRL+D**, **close**, **disconnect**, **status**, **get** (**mget**: mehrere Dateien downloaden), **put** (**mput**: mehrere Dateien übertragen), **hash** zeigt während der folgenden Datenübertragung nach jedem einzelnen Datenblock ein # Zeichen, **binary** od. **ascii** Format einstellen, **type** Übertragungsformat anzeigen, **cdup** cd.. Navigation
**Einstellungen**: **ntrans** Umwandlung einzelner Zeichen, **case** Groß/Kleinschreibung umwandeln, **nmap** Umwandlung nach festgelegten Regeln, **glob** Abschalten der diversen Umsetzungen, **runique** automatische eindeutige lokale Benennung einer Zieldatei (im Zweifelsfall mit fortlaufender Nummer), **sunique** remote
**Einstellungsdatei im Home Verzeichnis**: **.netrc**
**OPD Server**: Begriff im Zusammenhang mit mirrorn, heißt original point of distribution

## UUCP (UNIX to UNIX Communication Program)

→ bietet sich an, wenn keine "bessere" Netzverbindung zur Dateiübertragung vorhanden ist (keine gut ausgebauten Hard-/Softwareeigenschaften), benutzt das Spooling Verfahren

**Spooling Verfahren**: synonym **simultaneous peripherals operations on-line**, Zwischenspeicherung der Aufträge, das Programm **uucico** startet bei gefundenem Auftrag die Verbindung, nach Übertragungsbeendigung wird der Dämonprozess **uuxqt** zum Abarbeiten der Warteschlange gestartet
**UUCP System**: jeder Rechner besitzt einen vom Systemverwalter festgelegten UUCP Namen, **uuname** zeigt diese Namen an, default Zugriff immer auf das PUBDIR **/var/spool/uucppublic**, sofern durch den admin nicht modifiziert
**Benennung von Netzwerkshares**: **rechnername!~/verzeichnis/dateiname**
**Syntax**: **uucp kopieren_von kopieren_nach**
**Benutzerkommandos**: **uuto**, **uupick**, **uustat** Status des Benutzerauftrages, **uux** entfernte Auftragsbearbeitung auf einem direkt erreichbaren Host, **cu** Remote Benutzeranmeldung (nachgestellt wird der UUCP Rechnername oder eine Telefonnummer zum Anwählen, beginnend mit **0=** für Nebenstellenanlagen, **-** für WAIT), **~.** ESCAPE Sequenz zum Verbindungsende, **ct** Remote Rückruf auslösen
**Sicherheitseinschränkungen**: in **/etc/uucp/Permissions** zugelassene Kommandos, häufig zugelassen sind unter uux z.B. **lp**, **rmail**, **rnews**

### Elektronische Post

**Mailboxprinzip**: Benutzer haben eine **Primär-Mailbox** (Datei benannt mit Loginname in **/var/mail** bzw. **/usr/mail** bzw. **/usr/spool/mail**) und eine **Sekundär-Mailbox** (Name unter der Bourne-Shell Variablen **MAIL** einstellbar, default **mbox**)
**2 Formen der Mailadressierung unter UNIX**:

1. **UUCP**: **mailserver1!mailserver2!azrael!goutier**, die Mailserver sind die mediären Übertragungsserver
2. **Domain-Adressierung**: **goutier@azrael.moonshea.de**, innerhalb der Domäne Moonshea könnte man die Adressierung auch auf **goutier@azrael** verkürzen

**Mailheader**: Nachrichtenkopf
**Mailweiterleitung**: Anlegen der Datei **.forward** und Eintrag der Weiterleitungsadresse im Home Verzeichnis

### Diskussionsforum Netnews

**USENET**: eines der ältesten Computernetze, der dazugehörige Dienst heißt **Netnews**, Diskussionsforum mit sog. **Newsgroups**, Artikel werden "gepostet"
**Newsserver**: werden administriert und mit "Newsgroups" bestückt, das Angebot weicht bis auf wenige "zentrale Positionen" stark ab
→ früher basierte die Verbindung auf UUCP, heute auf NNTP
**Newsgroups**: hierarchisch aufgebaut, **mainstream groups**: comp, misc, news, sci, **andere Kategorien**: alt, de
→ einige Newsgroups werden moderiert, gepostete Artikel werden vom Newssystem per email an den Moderator gesendet, der über die Veröffentlichung entscheidet

## Grafische Benutzeroberflächen

### X11

→ als Projektteil "Athena" **1984** am MIT entstanden, logische Fortführung des Terminal Konzepts, betriebssystemunabhängiges Fenstersystem mit Client-Server Architektur, X-Terminal versteht Grafik-Primitivoperationen, wird vom **X-Konsortium** weiterentwickelt

**X-Server**: ist nicht an abgegrenzte Hardware gebunden sondern kann auch eine logische Abgrenzung sein z.B. ein Prozeß, der ein E/A Gerät (Display) bedient, besitzt weitreichende **Netzwerkfähigkeiten** (Hauptunterschied von X-Windows zu MS Windows), kann über TCP/IP an einen anderen X-Server umgeleitet werden, Benutzereingaben werden an Clients weitergeleitet, der X-Server weist Fenster zu
**X-Clients**: Applikationen, die das X-Window-System für ihre E/A Aktivitäten verwenden, erteilen dem Server entsprechende Aufträge

**X-Protokoll**: netzwerkfähige Sprache, über die sich X-Client und X-Server verstehen
**SaX**: **SuSE advanced X Configuration Tool**, SaX2 zur Konfiguration von **Xfree86 4.0**,

## Fenstermanager

→ ein spezieller X-Client
Übernimmt die Verwaltung von Fenstern (Größe / Anordnung etc.), der Windowmanager beherrscht sozusagen das Root-Fenster (voller Bildschirmbereich), Fenster sind hierarchisch, der Windowmanager befähigt zum Look&Feel
**COSE Initiative**: **common open software environment**, Einigung 1994, auf Motif aufbauend eine einheitliche Nutzeroberfläche für UNIX zu schaffen: **CDE common desktop environment**, Bausteine von Motif-Anwendungen nennt man **Widgets**,

## X-Windows-Operationen

→ zuerst wird der **X-Display-Manager xdm** gestartet, dieser startet den Fenstermanager
**Manueller Start im alphanumerischen Modus**: **startx** (Shellscriptstart wenn PATH Umgebungsvariablen vorhanden, falls unbekannt mit **xinit**) → Autostart beim Login ist in der **.profile** Datei unter **openwin**
**Mausoperationen**: beide äußere Tasten simulieren die Mitteltaste einer 3-Tastenmaus, Texte markieren mit Links/Rechtsklickkombinationen, Ziehen oder Mehrfachklick, Einfügen mit der Mitteltaste
**X-Client xterm**: Emulation eines VT102 Terminal

## X-Anwendungsstart auf entfernten Hosts

**Adressierung von X-Servern**: **Hostname:Servernummer.Bildschirmnummer** (sofern mehrere X-Server und Screens gehosted werden, ansonsten **Azrael:0.0**)
**Vorgang**: **rlogin azrael** → **DISPLAY=gargamel:0.0** → **export DISPLAY** → **xftp** oder **rlogin azrael** → **xftp –display gargamel:0.0** oder **rsh azrael xftp –display gargamel:0.0**
**Zugangskontrolle**: **hostbasiert** (**xhost** zeigt authorisierte Rechner an, **xhost +gargamel** Einzelbefugnis, **xhost +** globale Befugnis, **xhost –** Sperrung), **nutzerbasiert** (Zugriffscode in der Datei **.Xauthority** im Homeverzeichnis, beim Anmelden trägt **xdm** einen Zufallscode dort ein)
→ hostbasierte Zugangskontrolle hat immer Vorrang vor nutzerbasierter
**Sicherheitsanmerkung**: **Secure-Keyboard Option** aktivieren, damit Tastatureingaben nicht abgehört werden können
**NFS**: arbeitet mit einer globalen **.Xauthority**; sind jedoch mehrere Homeverzeichnisse auf unterschiedlichen Systemen vorhanden, muß mittels **xauth** der Zugangscode übertragen werden: **xauth extract – azrael:0 | rsh auf_server2 xauth merge -**

## KDE

**KDE**:

- Softwarebibliothek, um Programme mit einheitlicher Benutzerschnittstelle und Funktionalität zu schreiben
- Sammlung wichtiger Grundlagenprogramme wie z.B. KFM, KWM etc.

## KDE Hotkeys

| Hotkey | Funktion |
|---|---|
| STRG + T | Terminalfenster aus KFM aufrufen |
| ALT + F2 | Kommandoeingabefenster unter KDE aufrufen |
| STRG + ALT + ESC | Kill Freeze Windows (Totenkopf erscheint), **XkillClientEvent**, ESC wandelt den Totenkopfzeiger wieder in den normalen Mauszeiger zurück |
| STRG + ALT + ← | XServer und alle XProgramme beenden, LINUX Grundsystem und Netzprozesse laufen weiter! |
| ALT + linke Maustaste | Fenster an beliebiger Stelle ohne Titelleiste verschieben |
| ALT + TAB | Zwischen Anwendungen switchen |

**KDE Desktop**: besteht aus Windowmanager und Zusatzprogrammen
**Toolkits**: Sammlungen von Programmbausteinen, die für ein konsistentes Look and Feel für Anwendungen sorgen