Linux Essentials

LPI 010 Training

Vortragender: Jürgen Wurzer

Topics

- Linux Community & Open Source
- Befehlszeile und Arbeiten mit dem Dateisystem
- Komprimieren, Suchen, Textbearbeitung
- Shellskripts
- Grundbegriffe Hardware, Kernel, Prozesse
- Netzwerkgrundlagen
- Benutzer- und Gruppenkonten sowie Berechtigungen

Unix und Linux Geschichte und Versionen

* 1970: Unix (00:00:00, Jan 1, 1970 is time zero for UNIX) [Ken Thompson, Dennis Ritchie, Brian Kerni

… different implementation paths: System V, BSD

* 1982: GNU Project [Richard Stallman]

* 1988: Unix System V Release 4 (SVR4) [AT&T and SUN j

* 1991-94: Linux (an open source Unix Kernel) [Linus Torvalc

Unix Systems:

Solaris
AT&T Unix
SGI Unix
HP Ux
Ultrix,…….

Linux Distributions:

Linux Kernel + **GNU Tools** + Desktop-Tools + „PreConfiguration“

- Android und ChromeOS (Google)
- Red Hat (IBM), Fedora, CentOS(-Stream), OracleLinux
- SUSE (Novell), OpenSUSE
- Debian, Ubuntu, Kubuntu, Raspbian
- ….

BSD

Apple:
Mac OSx

Linux, Linux-Distributionen und seine vielseitigen Einsatzbereiche

- Welche Linux-Distributionen gibt es?
  - Debian, Ubuntu, Fedora, openSUSE, RHEL (Red Hat Enterprise Linux), CentOS, Linux Mint, …

- Was sind Linux Embedded Systeme?
  - minimalistische, auf spezifische Hardware-Funktion zugeschnittene
    Linux-Systeme die fest in einem Gerät verbaut sind.

- Wie werden Linux Embedded Systeme eingesetzt?
  - Alltagsgeräte: Router, Smart-TV, Fahrzeug-Infotainment, Smartphone (Android), Raspberry PI
  - Industriellen Steuerungen, Automobil-, Medizin- und Militäranwendungen

Linux, Linux-Distributionen und seine vielseitigen Einsatzbereiche

- Verschiedene Einsatzmöglichkeiten von Android
  - Android = spezielles Linux-System
  - Smartphones, Tablets, Smartwatches, Fahrzeugen, TV-Boxen

- Verschiedene Einsatzmöglichkeiten des Raspberry Pi
  - kostengünstiger, kreditkartengroßer Einplatinencomputer
  - Zum Programmieren lernen, als Mediacenter, kleiner Server (Webserver, Sambaserver), Bastelprojekte in der Hausautomatisierung, Retro-Gaming-System (z.B. RetroPie)

- Was ist Cloud Computing?
  - Bereitstellung von IT-Ressourcen wie Server, Speicher, Datenbanken und Software über das Internet (die „Cloud“) auf Abruf

- Welche Rolle spielt Linux beim Cloud Computing?
  - Ist das dominierende Betriebssystem in der Cloud
  - Meistens die Grundlage für Server, Container und Virtualisierungstechnologien

Linux Desktopanwendungen

- Office (LibreOffice oder Apache OpenOffice (beides Weiterentwicklungen von SUN StarOffice))
  - Writer, Calc, Impress, Draw, Base, Math

  → Entspricht: Word, Excel, PowerPoint, Visio, Access, Formeleditor
- Browser (Mozilla-Firefox, Google-Chromium)
- Email Client (Thunderbird)
- Multimedia-Anwendungen
  - Blender (3D-Grafiksoftware), GIMP (Bildbearbeitung), Inkspace (Vector-Grafik-Editor), Audacity (Audio-Editor), ImageMagic (Bildbearbeitung)
- Medienwiedergabe (VLC, smplayer, Audacious,….)

Linux Serverprogramme

- Webserver (Appache, Nginx, lightpd)
- Datenbank (MySQL, MariaDB, PostgreSQL, …)
- Datenfreigabe (NFS, Samba)
- Authentifizierungsserver (AD, Samba)
- Cloud-Lösungen (ownCloud, Nextcloud)
- Netzwerkdienste (DHCP, NTP, DNS, …)

- Programmiersprachen (Java, C, C++, JavaScript, Python, PHP, C#, Bash, Powershell)

Freie Software, OpenSource

- Freie Software (Richard Stallmann, 1985, GNU):

  sehr ideologisch, politisch, sozial:

  0. Ausführen wie man will und wozu man will
  1. Untersuchen und Anpassen wie man will
  2. Beliebige Weitergabe
  3. Beliebig Weiterentwickeln und wieder Freigabe (Pflicht!)

- Open Source

  Offener Quellcode, ev. mit Einschränkungen zur Verwendung

  → verschiedene Lizenzen

Lizenzen (FLOSS free/libre open soure sw)

- GPL (striktes Copyleft Prinzip)
  - Lizenz überträgt sich immer komplett auf Veränderungen
  - Anpassungen/Abschwächungen:
    - LGPL (Lesser GPL) erlaubt Mix mit unfreier SW
    - AGPL (Affero GPL) regelt Verkauf gehosteter SW
    - FDL (GNU Free Document License)
- OSI (OpenSource Initiative) permissive Prinzip
  - permissive = freizügig/erlaubend → max. Freiheit/Flexibilität
  - „OSI-Approved Lizenzen“
  - BSD (ganze Gruppe von Lizenzen, die fast nichts einschränken, OpenSource, den ich verändern, verwerten und dann auch als Closed Source vermarkten darf.
  - MIT-Lizenz, Apache-Lizenz, zlib-Lizenz, …
- CC Creative Commons
  - Open Source Prinzip für andere Bereiche

Desktop-Umgebungen

- Gnome (KISS, GTK, C)
- KDE (komplexer, Qt, C++)

- Terminalprogramme (Gnome Terminal, KDE Konsole, xterm, …)
- Virtuelle TTYs: Strg+Alt+F# (F1, F2, etc.)

- Präsentationssoftware (Libre-Impress, LaTeX-Beamer, Reveal.js)

- Projektmanagement-SW (GantProject, ProjectLibre)

Linux Server Einsatzgebiete

- Server
  - Webserver, Datenbank, Dateiserver, Mailserver
  - Netzwerkdienste (SSH, DNS, DHCP)
- Cloud
  - IaaS – Infrastructure as a Service
    - Virtuelle Hardware
    - Virtualisierung (Xen, KVM, VirtualBox)
  - PaaS – Platform as a Service
    - Fertige Umgebung / Platform
    - Heroku (Programmausführung ohne eigner VM)
  - SaaS – Software as a Service
    - Fertige Software über den Browser oder eine App
    - Dropbox, Salesforce (meist Browserinterface)

Datenschutz (beim Surfen)

- Cookie Tracking
- DNT (do not track) nur ein Wunsch!
- Privates Fenster
- Passwörter
  - Verwendung eines Passwort-Managers
  - KeePass (speichert verschlüsselt auf Platte)
  - BitWarden (speichert verschlüsselt in Cloud)
- Verschlüsselung
  - TLS/SSL
  - GnuPG
  - Festplattenverschlüsselung
    - Blockverschlüsselung (dm-crypt mit LUKS)
    - Stacked (EncFS, Veracrypt [gibt‘s auch auf Windows und Mac])

Unix/Linux-Nutzer lieben die Kommandozeile ;-)

Kommandozeile, Synonyme:

- command-line
- CLI = command-line interface
- Shell (Interpreter z.B: Bash, Zsh)
- Terminal (ist das Fenster)

Syntax Konvention für Kommandozeile:

Kommando Optionen Argumente

ls — *Optionen und Argumente sind optional*

*ls -l -s /etc /bin/*a** — *normalerweise Optionen zuerst*

ls -ls -a — *in einem Wort oder einzeln bzw. ls -lsa ls -l -s -a*

ls -- -la — *-- beendet Optionenliste!! ‚-la' ist hier ein Argument*

ls --all — *--longFormatOption* (benutzerfreundliche version z.B. eine Alternative zu -a verfügbar mit den meisten GNU tools)

Metazeichen & Quoting

- Metazeichen/Sonderzeichen haben eine besondere Bedeutung
  - `| < > & ; # * ? [ ] $ ~ ( ) \ ' "`

- Quoting
  - Aufhebung der Sonderbedeutung
  - Backslash \ hebt nächstes Sonderzeichen auf
    - Beispiel: `\*asd\\asd\'\$`
  - Zwischen Single Quotes ' hebt alles auf
    - Beispiel: `'asd"sad$asd*?'`
  - Zwischen Double Quotes ”
    - Beispiel: `"asd*?sd'as${hugo}as\$d`kom`sd"`

Variable u Kommando wird NICHT aufgehoben!

Shell Befehlstypen

- Shell Builtin
  - cd, pwd, echo, exit
- Funktion
  - Siehe Beispiel

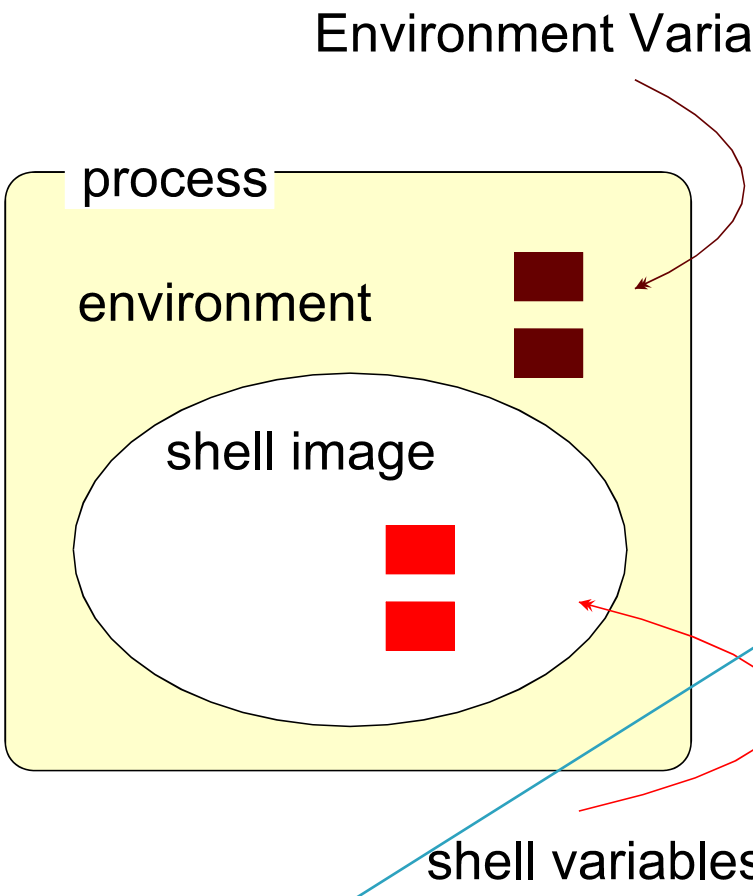
```
meine_funktion() {
  echo "Ich bin eine Funktion"
  ls -l
}

# Aufruf der Funktion
meine_funktion
```

- Alias
  - Kurzname bzw. Abkürzung für einen Befehl
  - Erzeugen mit: `alias Kurzname='Befehl'`
  - Anzeigen: `alias`
  - Löschen: `unalias Kurzname`
  - Dauerhaft: `~/.bashrc` oder `~/.bash_aliases`
- Hashed Kommando
  - Die shell merkt sich verwendete externe Kommandos in einer Hash-Tabelle
  - hash … Kommando zum ausgeben der Tabelle
- Externes Kommando
  - ls, cp, ln, …
  - Die meisten Kommandos sind extern

type Kommando

Gibt den Befehlstyp aus (extern oder buildin)

Variables and Environment



$ VARname=valuestring

$ export VARname
$ export VARname=value

$ Var=Val Command
$ TZ=EST date

export → wird in Umgebung des Prozesses aufgenommen
→ für Kindprozesse (Programme, Skripte, Sub-Shells) verfügbar

Variablen der BASH

- PATH
  - Suchpfad für Befehle
- HOME
  - Standard-Benutzerverzeichnis
- MAIL
  - Speicherort der E-Mails (Posteingang)
- MAILCHECK
  - Prüfintervall für neue E-Mails (in Sekunden)
- PS1
  - Primärer Shell-Prompt (z. B. $ oder #)
- PS2
  - Sekundärer Shell-Prompt (bei mehrzeiligen Befehlen)
- TERM
  - Terminal-Typ oder -Emulation (z. B. xterm)
- TZ
  - Zeitzoneneinstellungen des Systems

Auslesen mit $
z.B: echo $PATH

Variablen der BASH

- $$
  - Enthält die Prozess-ID (PID) der aktuellen Shell-Instanz
- $-
  - aktuelle Optionen der Shell. z. B. h für History-Erweiterung, i für interaktive Shell
- $!
  - Speichert die Prozess-ID (PID) des zuletzt im Hintergrund (&) gestarteten Befehls.
- $?
  - Enthält Exit-Status (Rückgabewert) des letzten Befehls
    - Wert 0 = Erfolg.
    - Wert ungleich Null (oft 1 bis 255) = Fehler

Hilfe unter Linux

- man
  - Zeigt die Handbuchseiten (Manpages) für Befehle an
- apropos (catman)
  - Sucht in den Beschreibungen aller Manpages nach einem bestimmten Stichwort. Dies ist sehr nützlich, wenn man den genauen Befehl nicht kennt.
- info
  - Zeigt die Dokumentation im Info-Format an, die oft detaillierter und besser verlinkt ist als Manpages.
- /usr/share/doc
  - Dies ist das Standardverzeichnis im Linux-Dateisystem, in dem zusätzliche Dokumentation (z. B. README-Dateien, Beispiele) für installierte Software-Pakete gespeichert wird.

Dateisuche unter Linux

- locate *pattern*
  - Findet Dateien schnell anhand eines Musters, indem es eine zuvor erstellte Datenbank durchsucht, anstatt das Dateisystem live zu durchsuchen
- updatedb
  - Aktualisiert Dateinamen-Datenbank. Normalerweise automatisch ausgeführt. DB wird von locate verwendet.
- find *verzeichnis*
  - Durchsucht das Dateisystem rekursiv nach Dateien
- find -name *pattern*
  - Durchsuchen mit Suchkriterium

Arbeiten mit Verzeichnisen

- Dateitypen
- Versteckte Dateien/Verzeichnisse
- $HOME, ~
- absolute und relative Pfade

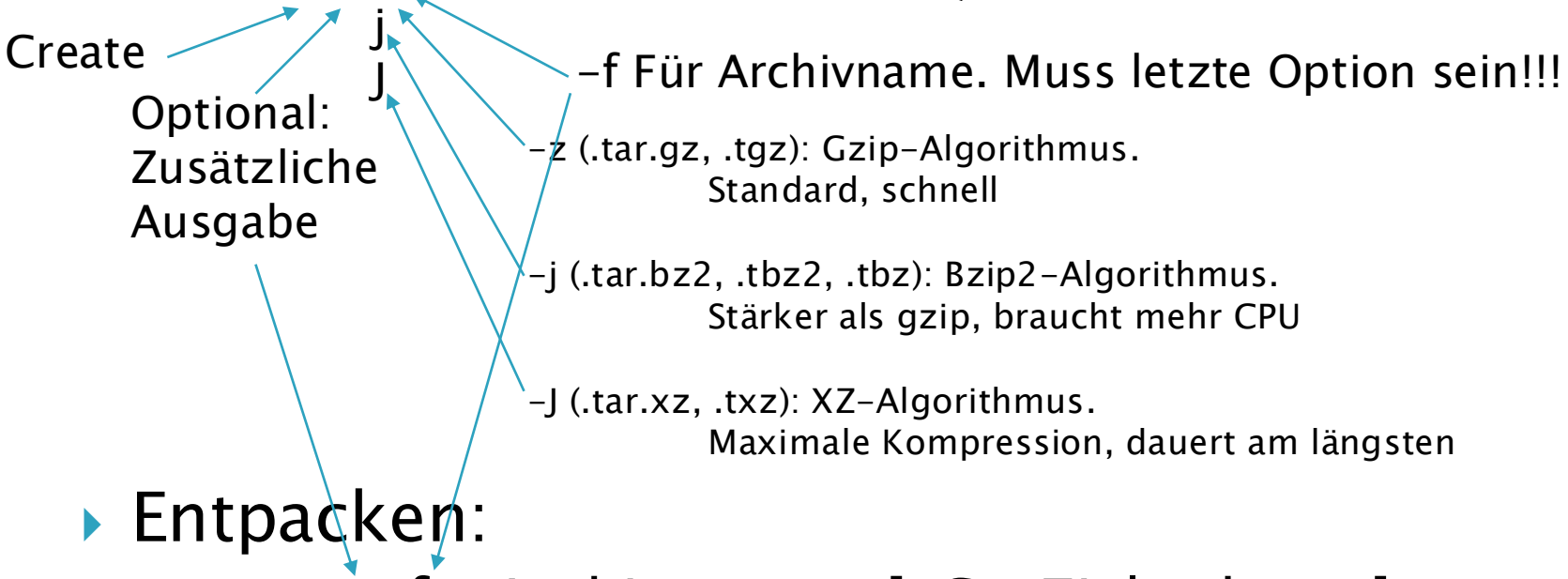
- Die wichtigsten Befehle + Optionen!
  - tree, ls, cd, pwd, mkdir, rmdir

Arbeiten mit Dateien

- Dateinamens-Regeln
- einfaches Globbing
  - * 0 oder mehr beliebige Zeichen
  - ? Genau ein beliebiges Zeichen
  - [1-4a-f]
  - [^1-4a-f]
  - [[:digit:]], [[:alpha:]], ……
- Befehle
  - cp, mv, rm, ln, touch, find

Dateien Archivieren und Komprimieren

- tar – Archiviert mehrere Dateien zusammen
- Erstellen:
  - tar -cvzf <Archivname> <Quellen>…



- Entpacken:
  - tar -xvzf <Archivname> [-C <Zielordner>]

Algo optional beim Entpacken → ohne → checkt Signatur

Dateien Komprimieren

- gzip, gunzip, bzip2, bunzip2, xz, unxz

- Komprimieren von datei.txt (wird dabei gelöscht)

```
gzip datei.txt          # Erstellt datei.txt.gz
bzip2 datei.txt         # Erstellt datei.txt.bz2
xz datei.txt            # Erstellt datei.txt.xz
```

- Dekomprimieren von datei.txt (komprimierte Datei wird gelöscht)

```
gunzip datei.txt.gz           # ODER: gzip -d datei.txt.gz
bunzip2 datei.txt.bz2         # ODER: bzip2 -d datei.txt.bz2
unxz datei.txt.xz             # ODER: xz -d datei.txt.xz
```

Komprimierte Tools

- zgrep, xzgrep, bzgrep
  - Text in komprimierten Dateien zu durchsuchen
  - .gz, .xz, .bz2 als auch .tar.gz, .tar.xz, .tar.bz2

- Beispiel: `zgrep "ERROR" /var/log/syslog.1.gz`
  - Optionen:
    - -n … Anzeigen der Zeilennummer
    - -i … Groß-/Kleinschreibung ignorieren

- zcat, xzcat, bzcat
  - Wie cat zu verwenden
  - Für komprimierte Dateien und Archive

Dateien Archivieren und Komprimieren

- zip, unzip (Windows ZIP Format)
- Packen:

```
# Komprimiert zwei spezifische Dateien in ein Archiv namens "projekt.zip"
zip projekt.zip datei1.txt datei2.pdf
# Komprimiert den gesamten Ordner "dokumente" inklusive aller Unterordner
zip -r dokumente_archiv.zip dokumente/
```

- Entpacken:

```
# Entpackt alle Dateien im aktuellen Ordner
unzip projekt.zip
# Entpackt den Ordner "dokumente" mit seiner Struktur im aktuellen Ordner
unzip dokumente_archiv.zip
```

Umleitung und Pipes

- 1> File
- 2> File
- &> File
- 1>> File
- 2>> File
- &>> File
- 0< File
- << hugo
  ....
  hugo
- <<< String

- Command | Command
- ls -l | head -20 | wc -w

- Filterprogramme:

cut, cat, find, less, head, tail
sort, wc, grep

Filterprogramme

cut: Extrahiert spezifische Spalten oder Zeichen aus Textzeilen.

- Beispiel: `cut -f 1 -d ',' datei.csv` (Zeigt die erste Spalte einer CSV-Datei an).

cat: Zeigt den gesamten Inhalt einer oder mehrerer Dateien an die Standardausgabe.

- Beispiel: `cat datei.txt`

find: Sucht nach Dateien und Verzeichnissen basierend auf Kriterien wie Name, Größe oder Typ.

- Beispiel: `find /home -name "*.jpg"` (Sucht alle JPG-Dateien im Ordner /home).

less: Ermöglicht das seitenweise Betrachten von langen Textdateien.

- Beispiel: `less grosse_datei.log` (Sie können mit Pfeiltasten scrollen und mit q beenden).

head: Zeigt standardmäßig die ersten 10 Zeilen einer Datei an.

- Beispiel: `head -n 5 datei.txt` (Zeigt nur die ersten 5 Zeilen).

tail: Zeigt standardmäßig die letzten 10 Zeilen einer Datei an.

- Beispiel: `tail -f /var/log/syslog` (Zeigt die letzten Zeilen einer Logdatei live an).

sort: Sortiert die Zeilen einer Textdatei alphabetisch oder numerisch.

- Beispiel: `sort unsortierte_liste.txt`

wc: Zählt Wörter, Zeilen und Zeichen in einer Datei.

- Beispiel: `wc -l datei.txt` (Zählt nur die Zeilenanzahl).

grep: Sucht nach einem bestimmten Textmuster oder regulärem Ausdruck innerhalb von Dateien.

- Beispiel: `grep "Fehler" logfile.txt` (Findet alle Zeilen mit dem Wort "Fehler").

Text durchsuchen mit Reguläre Ausdrücke

- grep = Text nach Mustern durchsuchen
- grep [Optionen] "Suchmuster" [Datei]

Ohne Datei von Standard Input

- Reguläre Ausdrücke (RegEx) für Suchmuster
  - 2 Modi: Basic (default) und Extended (-E)
- Basic:
  - Ohne Backslash für: . ^ $ * [ ] (mit \ normales Zeichen)
  - Mit Backslash für: ( ) | { } (ohne \ normales Zeichen)
- Extended:
  - Immer Spezialfunktion. Normales Zeichen mit \

Reguläre Ausdrücke

Symbol	Bedeutung	Beispiel	Treffer bei...
.	Ein beliebiges Zeichen	g.ep	grep, g-ep, g2ep
^	Zeilenanfang	^Hallo	Zeilen, die mit Hallo beginnen
$	Zeilenende	Ende$	Zeilen, die auf Ende enden
[ ]	Zeichenauswahl	[0-9]	Jede einzelne Ziffer
*	0 bis unendlich Mal	ab*	a, ab, abbb

Symbol	Bedeutung	Beispiel	Treffer bei...
?	0 oder 1 mal	App?le	"Aple" oder "Apple"
+	1 oder mehrmal	lo+l	"lol", "lool" (aber **nicht** "ll")
{n}	Genau n mal	[0-9]{3}	"123", "999" (genau 3 Ziffern)
|	Oder Verknüpfung	Hund|Katze	Hund oder Katze
( )	Gruppierung	(An)?fang (an|ab)bauen T(e|oa)st	"fang" oder "Anfang“ "anbauen" oder "abbauen” "Test" oder "Toast"

Einfache Skripts

- Ausführungsvarianten und nötige Rechte
- #! Shebang z.B. #!/bin/bash
- Argumente
  - $# = Anzahl der Argumente
  - $1, $2, … = einzelnen Argumente
  - $@ = Alle Argumente als eine einzige Zeichenfolge.
  - $* = Alle Argumente als separate, einzeln zitierte Zeichenfolgen.

Einfache Skripts

- if Bedingung – kann || bzw && enthalten

```
#!/bin/bash
FILE=$1
if [ -f "$FILE" ]; then
        echo "Datei '$FILE' existiert."
else
        echo "FEHLER: Datei '$FILE' nicht gefunden!"
fi
```

- exit [Statuscode]
  - Beendet Skript/Shell mit Statuscode (0 für Erfolg, 1–255 für Fehler)
  - Ohne Statuscode Argument wird $? verwendet
- for Schleife

```
# Iteriere über alle Argumente
for arg in "$@"; do
    echo "Argument ist: $arg"
done
```

Betriebssystemauswahl

- Vergleich mit Windows, Unix, Apple
- Linux Distributionen:
  - Enterprise Linux-Distributionen
    - Red Hat Enterprise Linux
    - ~~CentOS~~, Oracle Linux, Rocky Linux, AlmaLinux
    - SUSE Linux Enterprise Server
    - Debian GNU/Linux
    - Ubuntu LTS
  - Consumer Linux-Distributionen
    - Fedora
    - Ubuntu non-LTS
    - openSUSE
  - Experimentelle and Hacker-Linux-Distributionen
    - Arch
    - Gentoo

Hardware Grundlagen und Befehle

- **`lshw`** – Listet die gesamte Hardware auf. sudo lshw → noch mehr Details
- **`lscpu`** – Informationen über die CPU
- **`lsblk`** – Zeigt Informationen zu allen Blockgeräten (Festplatten, Partitionen, USB-Sticks)
- **`lspci`** – Listet alle angeschlossenen PCI-Bus Geräte auf.
- **`lsusb`** – Details zu angeschlossenen USB-Geräten
- **`free -h`** – Zeigt die Speicherauslastung (freien und belegten RAM sowie Swap-Speicher)
- **`df -h`** – Zeigt den freien Speicherplatz aller eingehängten Laufwerke an ("disk free").
- **`uname -a`** – Zeigt grundlegende Systeminformationen an, einschließlich des Kernel-Namens und der Version.
- **`dmidecode`** – Liest Hardwareinformationen direkt aus der DMI-Tabelle (BIOS/UEFI) aus (erfordert sudo).

Dateisystemaufbau

Programmverzeichnisse:

$ which cp

- /bin
- /sbin
- /usr/bin
- /usr/sbin
- /usr/locale/bin
- /usr/locale/sbin
- ~/bin
- ~/.local/bin

~/.*

- .profile, …..

/etc:

„Systemconfig-DB“

- passwd
- shadow
- group
- hostname
- hosts
- profile, bash.bashrc
- nanorc
- resolv.conf
- sysctl.conf
- logrotate.conf
- crontab
- apt/sources*, …

*oder Verzeichnisse dafür: *.d*

Dateisystemaufbau

- Teile, die den Kernel ausmachen
  - /boot/* und /lib/modules/Version/*
- GRUB
- systemd oder init Konfigurationen
- /proc !!!
- /dev !!!
- Speicher und SWAP

Prozesse

- top
- ps
- /proc/PID
- uptime
- syslog und die wichtigsten Dateien in /var/log
- dmesg (Kenrel-RingBuffer)
- systemd-journald: journalctl

Netzwerk Grundkonfiguration abfragen

- Routing und IP Adressen anzeigen
  - ip route show
  - ip addr show
- Netwerk Interface konfigurieren
  - sudo ip link set dev eth0 up
  - sudo ip addr add 192.168.1.100/24 dev eth0
- netstat, ss
  - Informationen zu TCP-, UDP- und Unix-Sockets sowie offene Ports und aktive Verbindungen an
- DNS Client Konfiguration
  - /etc/resolv.conf, /etc/hosts
- ping
  - Testet Erreichbarkeit eines Netzwerkziels
- host
  - `host google.com #Zeigt die IPv4- und IPv6-Adressen der Domain an.`
  - `host 8.8.8.8  #Findet den Hostnamen heraus, der zu dieser IP gehört.`

Sicherheit in Linux – Benutzerkonten Grundprinzip

- UID/GUI (Nummern sehr Distro-abhängig)
  - 0: root
  - <100 oder <500: System Accounts
  - >1000 Standarduser und Service Accounts
- Dateien
  - /etc/passwd, /etc/shadow
  - /etc/group, /etc/gshadow
  - /etc/sudoers, /etc/sudoers.d
- Befehle:
  - passwd, id, who, w, sudo, chsh, chfn

Sicherheit in Linux – Benutzerkonten Grundprinzip

- Befehle:
  - passwd
    - Ändern des Passworts
  - id
    - User-ID (UID) und die Gruppen-IDs (GID) anzeigen
  - who
    - Listet alle Benutzer auf, die aktuell am System angemeldet sind.
  - w
    - Erweitert who. Tätigkeit der Benutzer (welche Prozesse sie ausführen)
  - chsh
    - Standard-Login-Shell konfigurieren
  - chfn
    - Ändern von hinterlegten Benutzerinformationen

Sicherheit in Linux – Benutzerkonten Grundprinzip

- Befehle:
  - sudo
    - Erlaubt es berechtigten Benutzern, Befehle mit den Rechten eines anderen Benutzers auszuführen – meistens mit den Rechten des Administrators (Root).
  - Benutzer hinzufügen:
  - Unter den meisten Distributionen (Ubuntu, Debian) reicht es aus, den Benutzer der Gruppe sudo hinzuzufügen.
    - sudo usermod -aG sudo BENUTZERNAME
  - Ansonst per visudo bearbeiten
    - Niemals direkt /etc/sudoers bearbeiten

Benutzer verwalten

- /etc/passwd, /etc/shadow, /etc/group, /etc/skel/
- useradd, groupadd
- userdel, groupdel
- passwd

- adduser, addgroup
  - Verfügbar bei Debian-basierten Distributionen
  - Benutzerfreundliche Wrapper für useradd und groupadd

- Beziehungsweise per GUI

Dateiberechtigungen

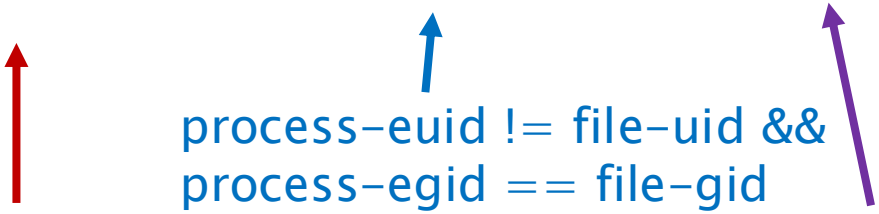
Alles ist eine Datei! → Benötigen nur Dateiberechtigungen!

Permission Dateirechte	File Datei	Directory Verzeichnis	Device / Spec. File Gerätedatei
r	read, copy	read (name + InodeNr)	read
w	write, change	write (create, rename, delete)	write
x	execute	use Inodes of dir-entries (what can you do without??)	---

Standard Permissions

Standard Zugriffsrechte

user	group	other
r w x	r w -	r - -
1 1 1	1 1 0	1 0 0
7	6	4



process-euid == file-uid

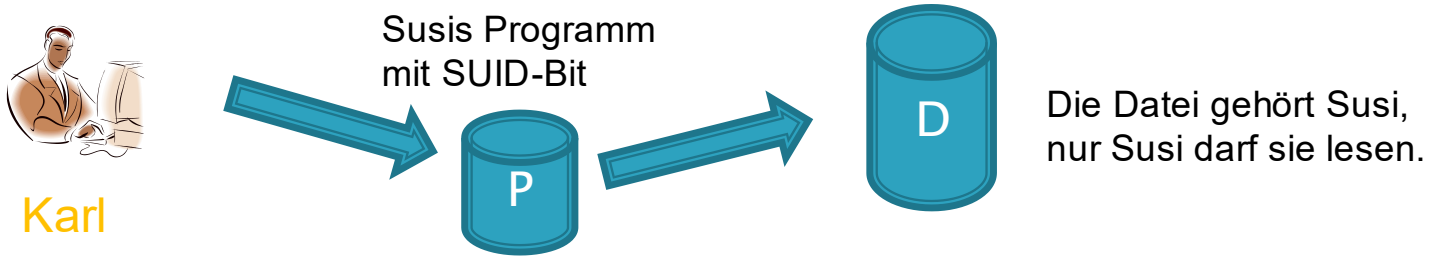
process-euid != file-uid &&
process-egid == file-gid

euid != file-uid &&
egid != file-gid

SetUID-Bit: Rechte für Sonderaktionen durch Wechsel der effektiven Benutzer-ID

Karl ruft ein Programm auf, das *Susi* gehört und bei dem das Set-User-ID-Bit gesetzt ist

→ wenn Karl nun eine Datei öffnet, die Susi gehört, wird Susi als zugreifende Identität verwendet.



Wenn der Besitzer eines „SUID-Programms“ root ist, erlaubt der Betriebssystemkern alles. Damit verlagert sich die gesamte Verantwortung für die Sicherheit vom Kernel-Code auf den Programmcode!

Spezielle Permission Bits auf Verzeichnisse (FS abhängig)

- Sticky bit
  - Mit Sticky Bit: Jeder kann zwar Dateien erstellen, aber nur der Besitzer der Datei (oder root) darf diese löschen oder umbenennen.
- SetGUID Bit
  - Mit SetGID-Bit: Jede neue erstellte Datei und Unterverzeichnis, in diesem Ordner, erhält automatisch dieselbe Gruppe wie das Verzeichnis selbst – egal, welcher User die Datei erstellt.

Access Control Lists (ACL/DACL)

sind in den meisten FS-Implementierungen verfügbar; Unix-Leute benutzen sie meistens nicht!

```
# e.g. POSIX ACLs:
$ setfacl m:775,u:hans:770 file1 file2 ….
$ setfacl -m g:programmers:rx file1 file2 ….
$ getfacl file1 file2 ….
```

Filesystem Kommandos

- ls
- pwd
- cd
- mkdir
- rmdir
- du
- df

- cp
- mv
- ln
- rm
- cat

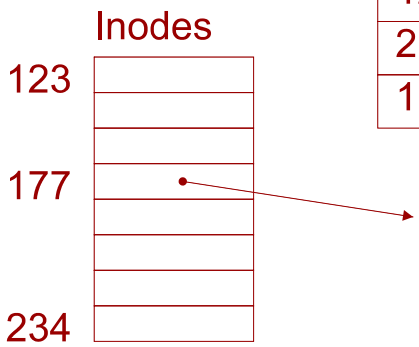
- more, pg, less
- pr
- file
- find
- touch

- chmod
- chown
- chgrp

Hard-Link

directory

123	name_1
123	name_5
123	name_6



Symbolic-Link

directory

123	name_1
234	name_2
177	name_3

datafile

/path.../name_2

INODE speichert:
Dateityp, Zugriffsrechte (rwx), GID, UID,
Dateigröße, Hardlinkzähler,
Verweis auf die Datenblöcke;
Dateiname ist NICHT enthalten!