

Betriebssysteme

SS 2015

Hans-Georg Eßer

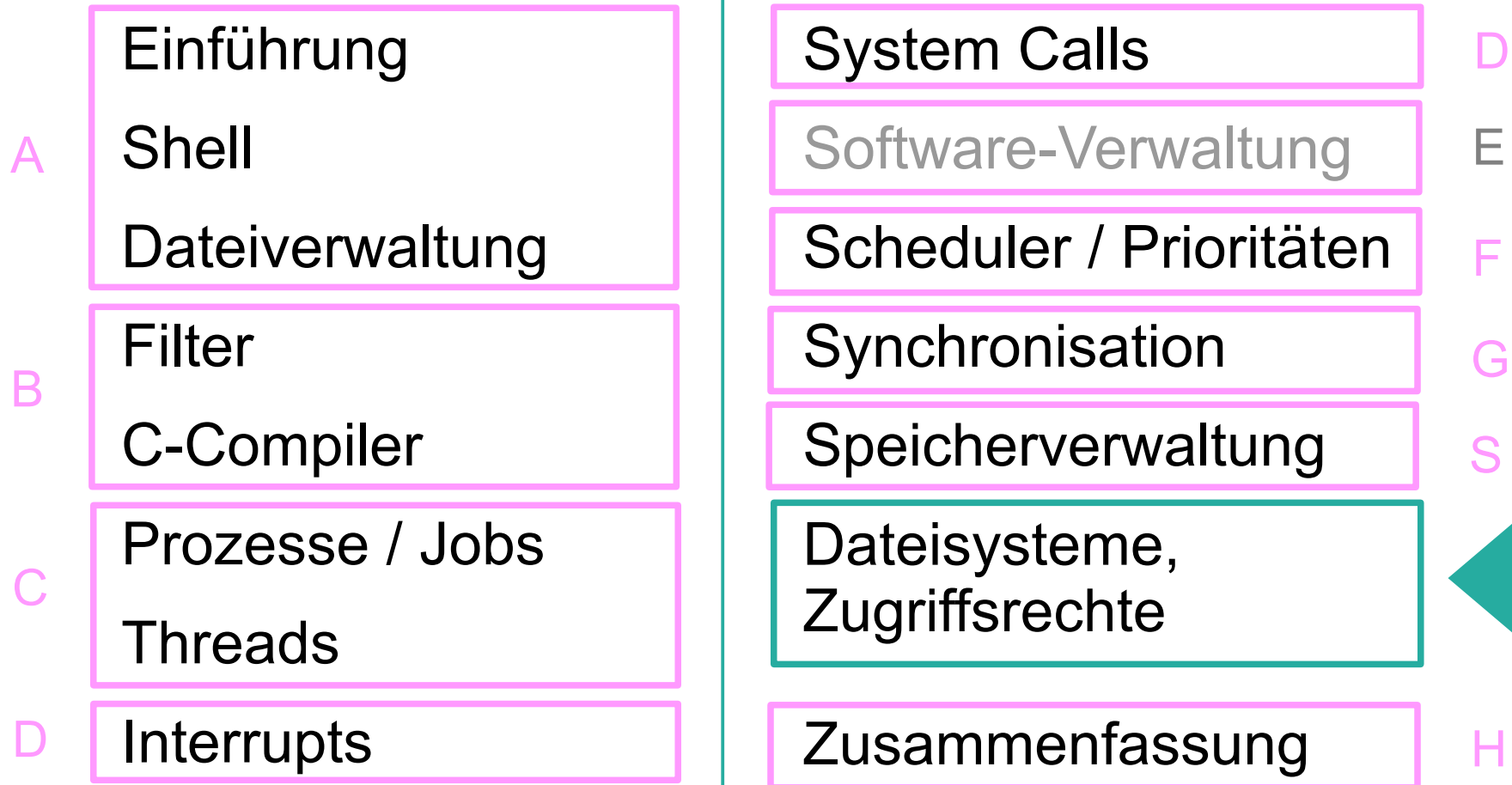
Dipl.-Math., Dipl.-Inform.

Foliensatz T:

- Dateisysteme
- Zugriffsrechte

v1.0, 2015/04/27

Übersicht: BS Praxis und BS Theorie



Dateisysteme

- Linux (und andere BS) unterteilen Festplatten in Partitionen
 - traditionell: vier Partitionen
 - Anfang, Ende, Größe: in Partitionstabelle im MBR (Master Boot Record)
 - Bezeichnung: **primäre Partitionen**
 - falls mehr nötig: eine der vier Partitionen zur **erweiterten** Partition machen
 - darin: **logische Partitionen**

- Windows vergibt für jede (Windows)-Partition einen Laufwerksbuchstaben (C:, D: etc.)
 - unabhängig von Status primär/logisch
 - Reihenfolge kann wechseln
- Linux verwendet Bezeichnungen, die sich aus
 - Typ der Platte (IDE, SCSI)
 - Gerätenummer
 - Partitionsnummer
 zusammensetzen (sda1 = SCSI disk a, part. 1)

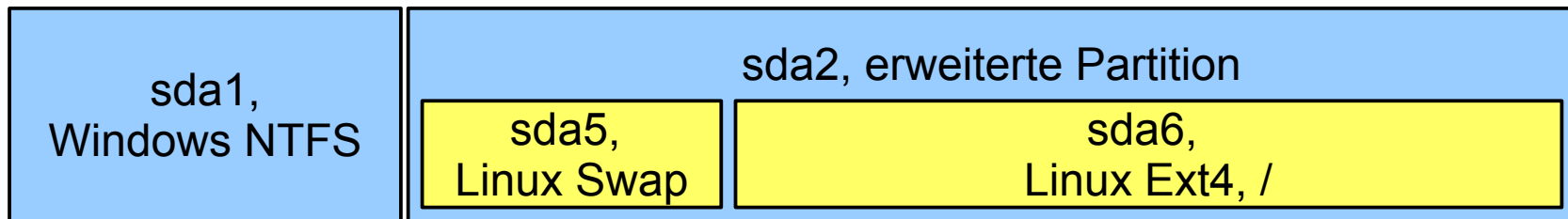
- Festplatten
 - sda, sdb, sdc, ...: SCSI und moderne SATA
 - hda, hdb, hdc, ...: klassische IDE
- Partitionen
 - 1, 2, 3, 4: primäre Partitionen
 - 5, 6, 7, ...: logische Partitionen (dann muss mind. eine der primären Part. eine erweiterte sein)
- Zugriff über Gerätedateien:
 - sda3 → /dev/sda3

- Gerätedateien erzeugen moderne Linux-Versionen dynamisch:

```
esser@dissdevel:~$ ls -l /dev/sd*  
brw-rw---- 1 root disk 8, 0  2. Jun 17:15 /dev/sda  
brw-rw---- 1 root disk 8, 1  2. Jun 17:15 /dev/sda1  
brw-rw---- 1 root disk 8, 2  2. Jun 17:15 /dev/sda2  
brw-rw---- 1 root disk 8, 5  2. Jun 17:15 /dev/sda5
```

- in alten Linux-Versionen: große Mengen an passenden Gerätedateien statisch erzeugt

- Typische Partitionierung



- sda1: 1. primäre Partition: Windows, NTFS („Laufwerk C:“)
- sda2: erweiterte Partition, enthält logische
- sda5: 1. logische Partition: Linux, Swap
- sda6: 2. logische Partition: Linux, Ext4

- Arbeiten mit Gerätedateien
 - `head /dev/sda1`
gibt Anfang der Partition sda1 aus
 - `dd if=/dev/sda1 of=/tmp/image.dat`
erzeugt 1:1-Kopie der Partition sda1 in Datei,
if=input file, of=output file
 - `fdisk /dev/sda`
bearbeitet Partitionstabelle der Festplatte sda
 - `mkfs.ext3 /dev/sda7`
formatiert Partition sda7 mit Ext3-Dateisystem

- Partitionieren unter Linux

- `fdisk`: Standard-Tool
- `cdisk`: „grafisches“ Tool

```

xterm
cfdisk 2.12q
Festplatte: /dev/sda
Größe: 300090728448 Bytes, 300,0 GB
Köpfe: 255  Sektoren pro Spur: 63  Zylinder: 36483

Name      Flags      Part. Typ  Dateisystemtyp  [Bezeichner]  Größe (MB)
-----
sda1      Boot       Primäre   NTFS             [^L]          103318,72*
          Logische   Freier Bereich
sda5      Logische   Linux swap / Solaris
sda6      Logische   Linux ReiserFS
          Logische   Freier Bereich
sda3      Primäre   M95 FAT32 (LBA)
sda4      Primäre   Hidden M95 FAT32 (LBA)

[ Bootbar ] [ Löschen ] [ Hilfe ] [ Maxim. ] [ Ausgabe ]
[ Ende ] [ Typ ] [ Einheit. ] [ Schreib. ]

(De)Aktivieren des bootfähig-flags der aktuellen Partition
  
```

- `sfdisk`: für Skript-gesteuertes Partitionieren

Partitionsliste anzeigen

```
server:~# fdisk -l
```

```
Disk /dev/sda: 10.7 GB, 10694426624 bytes
255 heads, 63 sectors/track, 1300 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x000ce798
```

Device	Boot	Start	End	Blocks	Id	System
/dev/sda1	*	1	1241	9965568	83	Linux
/dev/sda2		1241	1301	475137	5	Extended
/dev/sda5		1241	1301	475136	82	Linux swap / Solaris

Platte partitionieren

```
server:~# fdisk /dev/sda
```

```
Command (m for help): _
```

- `p` – zeigt die Partitionstabelle (wie in `fdisk -l /dev/sda`).
- `n` – legt eine neue Partition an; fragt Partitionstyp, Nummer der Partition und Größe ab.
- `t` – Ändert den Typ einer Partition. Nach dem Aufruf des Kommandos erhalten Sie mit dem Kommando `l` eine Übersicht über die `fdisk` bekannten Partitionstypen.
- `d` – Löscht eine Partition.
- `w` – schreibt die von Ihnen überarbeitete Partitionstabelle. Danach beendet sich `fdisk`.
- `q` – Programm beendet sich, ohne die Partitionstabelle zu ändern.
- `m` – Menü, in dem alle Befehle aufgeführt sind, nur in Englisch und noch ein paar mehr.

Neue primäre Partition erzeugen

```
Command (m for help): n
Command action
  e   extended
  p   primary partition (1-4)
p
Partition number (1-4): 2
First cylinder (1241-1300, default 1241):
Using default value 1241
Last cylinder, +cylinders or +size{K,M,G} (1241-1300, default 1300):
Using default value 1300
```

```
Command (m for help): p
```

```
Disk /dev/sda: 10.7 GB, 10694426624 bytes
255 heads, 63 sectors/track, 1300 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x000ce798
```

Device	Boot	Start	End	Blocks	Id	System
/dev/sda1	*	1	1241	9965568	83	Linux
/dev/sda2		1241	1300	475658	83	Linux

Neue erweiterte Partition erzeugen

```
Command (m for help): n
Command action
  e   extended
  p   primary partition (1-4)
e
Partition number (1-4): 2
First cylinder (1241-1300, default 1241):
Using default value 1241
Last cylinder, +cylinders or +size{K,M,G} (1241-1300, default 1300):
Using default value 1300
```

```
Command (m for help): p
```

```
Disk /dev/sda: 10.7 GB, 10694426624 bytes
255 heads, 63 sectors/track, 1300 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x000ce798
```

Device	Boot	Start	End	Blocks	Id	System
/dev/sda1	*	1	1241	9965568	83	Linux
/dev/sda2		1241	1300	475658	5	Extended

Neue logische Partition erzeugen

```
Command (m for help): n
Command action
  l   logical (5 or over)
  p   primary partition (1-4)
l
First cylinder (1241-1300, default 1241):
Using default value 1241
Last cylinder, +cylinders or +size{K,M,G} (1241-1300, default 1300): 1260
```

Auswahl geändert, weil es
jetzt eine erweiterte
Partition gibt!



```
Command (m for help): p
```

```
Disk /dev/sda: 10.7 GB, 10694426624 bytes
255 heads, 63 sectors/track, 1300 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x000ce798
```

Device	Boot	Start	End	Blocks	Id	System
/dev/sda1	*	1	1241	9965568	83	Linux
/dev/sda2		1241	1300	475658	5	Extended
/dev/sda5		1241	1260	154326+	83	Linux

```
Command (m for help): t
Partition number (1-5): 1
Hex code (type L to list codes): L
```

Partitionstypen

0	Empty	24	NEC DOS	81	Minix / old Lin	bf	Solaris
1	FAT12	39	Plan 9	82	Linux swap / So	c1	DRDOS/sec (FAT-
2	XENIX root	3c	PartitionMagic	83	Linux	c4	DRDOS/sec (FAT-
3	XENIX usr	40	Venix 80286	84	OS/2 hidden C:	c6	DRDOS/sec (FAT-
4	FAT16 <32M	41	PPC PReP Boot	85	Linux extended	c7	Syrinx
5	Extended	42	SFS	86	NTFS volume set	da	Non-FS data
6	FAT16	4d	QNX4.x	87	NTFS volume set	db	CP/M / CTOS / .
7	HPFS/NTFS	4e	QNX4.x 2nd part	88	Linux plaintext	de	Dell Utility
8	AIX	4f	QNX4.x 3rd part	8e	Linux LVM	df	BootIt
9	AIX bootable	50	OnTrack DM	93	Amoeba	e1	DOS access
a	OS/2 Boot Manag	51	OnTrack DM6 Aux	94	Amoeba BBT	e3	DOS R/O
b	W95 FAT32	52	CP/M	9f	BSD/OS	e4	SpeedStor
c	W95 FAT32 (LBA)	53	OnTrack DM6 Aux	a0	IBM Thinkpad hi	eb	BeOS fs
e	W95 FAT16 (LBA)	54	OnTrackDM6	a5	FreeBSD	ee	GPT
f	W95 Ext'd (LBA)	55	EZ-Drive	a6	OpenBSD	ef	EFI (FAT-12/16/
10	OPUS	56	Golden Bow	a7	NeXTSTEP	f0	Linux/PA-RISC b
11	Hidden FAT12	5c	Priam Edisk	a8	Darwin UFS	f1	SpeedStor
12	Compaq diagnost	61	SpeedStor	a9	NetBSD	f4	SpeedStor
14	Hidden FAT16 <3	63	GNU HURD or Sys	ab	Darwin boot	f2	DOS secondary
16	Hidden FAT16	64	Novell Netware	af	HFS / HFS+	fb	VMware VMFS
17	Hidden HPFS/NTF	65	Novell Netware	b7	BSDI fs	fc	VMware VMKCORE
18	AST SmartSleep	70	DiskSecure Mult	b8	BSDI swap	fd	Linux raid auto
1b	Hidden W95 FAT3	75	PC/IX	bb	Boot Wizard hid	fe	LANstep
1c	Hidden W95 FAT3	80	Old Minix	be	Solaris boot	ff	BBT
1e	Hidden W95 FAT1						

- Einfaches Anlegen einer neuen Partition macht diese noch nicht benutzbar
- Partition muss man vor erster Nutzung **formatieren** (= mit einem **Dateisystem** versehen)
- Kommando allgemein: `mkfs` (make filesystem)
 - `mkfs -t TYP /dev/GERÄT`
 - ruft spezialisiertes Tool `mkfs.TYP` (z. B. `mkfs.ext3`) auf

```
root@dissdevel:/# ls /sbin/mkfs*
/sbin/mkfs           /sbin/mkfs.ext2      /sbin/mkfs.ext4dev  /sbin/mkfs.ntfs
/sbin/mkfs.bfs       /sbin/mkfs.ext3      /sbin/mkfs.minix    /sbin/mkfs.vfat
/sbin/mkfs.cramfs    /sbin/mkfs.ext4      /sbin/mkfs.msdos
```

```

root@dissdevel:/# mkfs -t ext3 /dev/sda8
mke2fs 1.41.12 (17-May-2010)
Dateisystem-Label=
OS-Typ: Linux
Blockgröße=1024 (log=0)
Fragmentgröße=1024 (log=0)
Stride=0 Blöcke, Stripebreite=0 Blöcke
2560 Inodes, 10240 Blöcke
512 Blöcke (5.00%) reserviert für den Superuser
Erster Datenblock=1
Maximale Dateisystem-Blöcke=10485760
2 Blockgruppen
8192 Blöcke pro Gruppe, 8192 Fragmente pro Gruppe
1280 Inodes pro Gruppe
Superblock-Sicherungskopien gespeichert in den Blöcken:
    8193

```

```

Schreibe Inode-Tabellen: erledigt
Erstelle Journal (1024 Blöcke): erledigt
Schreibe Superblöcke und Dateisystem-Accountinginformationen: erledigt

```

Das Dateisystem wird automatisch nach jeweils 30 Einhäng-Vorgängen bzw. alle 180 Tage überprüft, je nachdem, was zuerst eintritt. Dies kann durch `tune2fs -c` oder `-i` geändert werden.

- Auch **Swap-Partition** (Bereich, der für das Auslagern von Speicherseiten verwendet wird; → **Paging**) muss formatiert werden
- Tool heißt `mkswap`:

```
root@dissdevel:/# mkswap /dev/sda5  
Setting up swapspace version 1, size = 475132 KiB  
no label, UUID=5c43f2b7-8801-4fde-94a2-f154ffbabb42
```
- Swap-Bereich darf auch Datei sein
→ hilfreich, wenn keine Swap-Partition angelegt werden kann

- Linux bindet beim Systemstart nicht automatisch alle Dateisysteme (meist: Partitionen) ein, sondern tut dies nur für eine Auswahl, die durch Einträge in einer Konfigurationsdatei festgelegt wird. Ausnahme: Root-Dateisystem /, ohne das kein Systemstart möglich ist.
- Den Einbindevorgang nennt Linux (wie alle Unix-Systeme) **mounten**, die umgekehrte Operation, bei der das System nicht länger auf einen Datenträger zugreift, heißt **unmounten**.
- Die dafür zuständigen Kommandos heißen `mount` und `umount` (nicht `unmount`!)
- Automatisches Mounten über Einträge in `/etc/fstab`

- Das Mounten stellt eine Verknüpfung zwischen einem Datenträger und einem Verzeichnis her, unter dem dann die Inhalte des Datenträgers erreichbar sind
- Diese Verzeichnisse (**Mount-Points**) sind das Gegenstück zu Windows-Laufwerksbuchstaben
- Linux- (Unix-) Ansatz ist flexibler

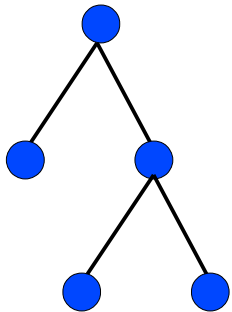
Datenträger unter Windows und Linux

Linux-Partition: nicht sichtbar	[Root-Dateisystem /dev/sda6 auf /] /home /usr /etc /var ...
C: [Win] C:\Windows C:\Windows\System C:\Users C:\Users\Esser C:\Users\Esser\Documents	[/dev/sda1 auf /mnt/win1] /mnt/win1/Windows /mnt/win1/Windows/System /mnt/win1/Users /mnt/win1/Users/Esser /mnt/win1/Users/Esser/Documents
D: [Restore] D:\Restore.Tmp	[/dev/sda2 auf /mnt/win2] /mnt/win2/Restore.Tmp
E: [OfficeDVD] E:\Files	[Office-DVD auf /media/OfficeDVD] /media/OfficeDVD/Files

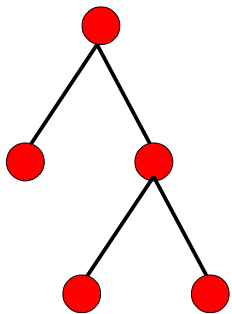
Mounten (4)

Windows

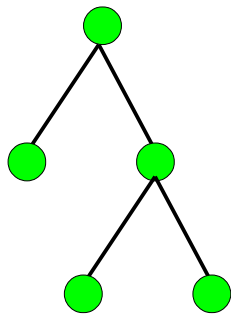
C:



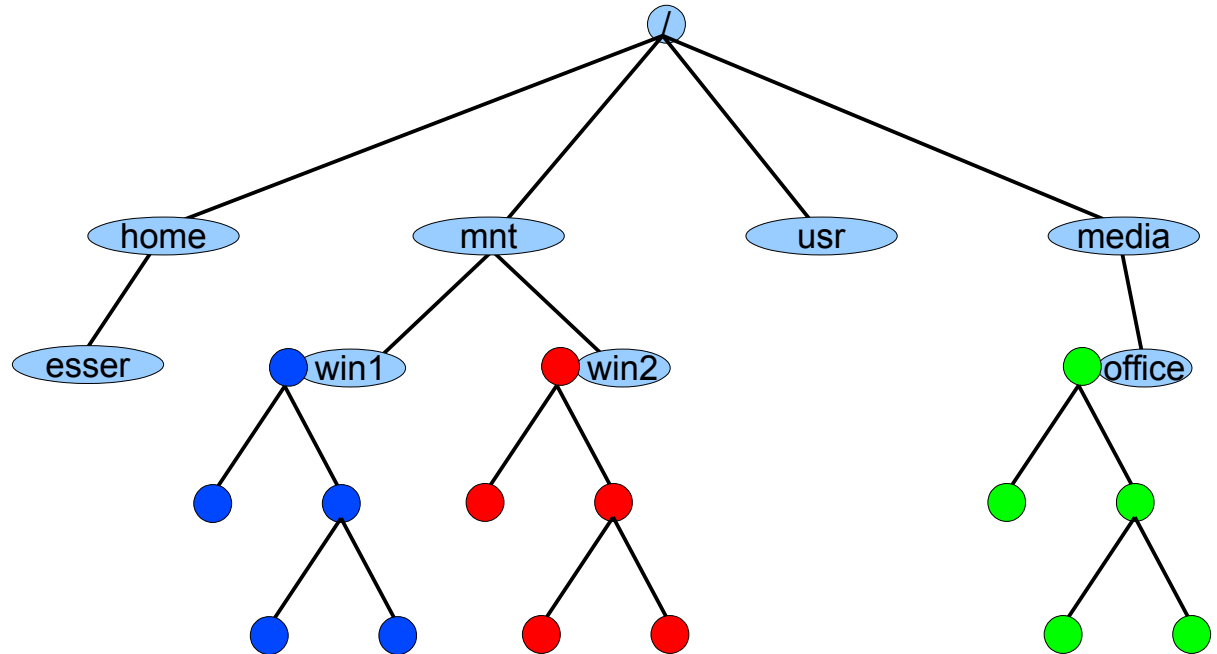
D:



E:



Linux



- Was braucht man fürs Mounten?
 - Gerätedatei des Datenträgers (Partition o. ä.)
 - Mount-Point (Verzeichnis, muss schon existieren)
 - evtl. Typ des Dateisystems
 - evtl. Optionen fürs Mounten

mount

```

-t TYP                -o OPTIONS
/dev/PARTITION      /MOUNTPOINT

```

```
mount -t ext3 -o ro /dev/sda7 /mnt
```


- Dateisystemtyp (`-t TYP`)
 - `ext4`: 4th extended filesystem (Linux, aktuell)
 - `ext3`: 3rd extended filesystem (Linux, älter)
 - `ext2`: 2nd extended filesystem (Linux, veraltet)
 - `reiserfs`: Reiser-Dateisystem (Linux, älter)
 - `ntfs`: New Technology Filesystem (Windows)
 - `vfat`: Virtual File Allocation Table (DOS, Windows)
 - `iso9660`: CD-/DVD-Dateisystem
 - `udf`: DVD-Dateisystem (z. B. Video-DVD)

- Dateisystemtyp (`-t TYP`)
 - Liste tatsächlich noch länger; Auszug aus Manpage:

`-t, --types vfstype`

The argument following the `-t` is used to indicate the filesystem type. The filesystem types which are currently supported include: `adfs, affs, autofs, cifs, coda, coherent, cramfs, debugfs, devpts, efs, ext, ext2, ext3, ext4, hfs, hfsplus, hpfs, iso9660, jfs, minix, msdos, ncpfs, nfs, nfs4, ntfs, proc, qnx4, ramfs, reiserfs, romfs, squashfs, smbfs, sysv, tmpfs, ubifs, udf, ufs, umsdos, usbfs, vfat, xenix, xfs, xiafs`.

- Welche Dateisysteme unterstützt der Kernel (im Moment)?

```
root@dissdevel:~# grep -v nodev /proc/filesystems
ext3
fuseblk
udf
iso9660
ntfs
vfat
```

- Mount-Optionen (`-o` *OPTIONEN*); Auswahl:
 - `ro`: read-only (nur lesen)
 - `rw`: read-write (lesen und schreiben; Standard)
 - `async`, `sync`: alle Zugriffe asynchron bzw. synchron (sofort schreiben, kein Puffer) ausführen
 - `noatime`: Zugriffe auf Dateien nicht in Metadaten speichern (u. a. für Flash-Datenträger sinnvoll)
 - `nodiratime`: wie `noatime`, für Verzeichnisse
 - `noexec`: Programme sind nicht ausführbar
 - `remount`: bereits gemountetes FS nochmal mounten
 - `loop`: Dateisystem-Image mounten

- Swap-Partitionen werden nicht gemountet, sondern aktiviert (`swapon`) oder deaktiviert (`swapoff`)

```
root@dissdevel:/# swapon -v /dev/sda5  
swapon on /dev/sda5  
swapon: /dev/sda5: found swap signature: version 1, page-size 4,  
      same byte order  
swapon: /dev/sda5: pagesize=4096, swapsize=486539264,  
      devsize=486539264
```

```
root@dissdevel:/# swapoff -v /dev/sda5  
swapoff on /dev/sda5
```

- (ohne Option `-v` keine Ausgabe)
- Swap darf auch eine Datei sein

- Übersicht über aktive Swap-Bereiche

```
root@dissdevel:/# cat /proc/swaps
Filename      Type      Size      Used      Priority
/dev/sda5     partition 475128    0         -1
/tmp/swapfile file       10232     0         -2
```

- Dateisystem wieder aushängen (unmounten)
 - Kommando `umount`
 - Argument: Wahlweise Name der Gerätedatei (`/dev/...`) oder Mount-Point
 - Beispiele:
 - `umount /dev/sda6` (Gerätedatei)
 - `umount /mnt/win1` (Mount-Point)

- Kommando `umount` schlägt manchmal fehl:

```

root@dissdevel:/mnt/tmp# pwd
/mnt/tmp
root@dissdevel:/mnt/tmp# umount /mnt
umount: /mnt: device is busy.
        (In some cases useful info about processes that use
         the device is found by lsof(8) or fuser(1))
root@dissdevel:/mnt/tmp# cd /
root@dissdevel:/# umount /mnt/
root@dissdevel:/# _

```

- Es darf keine Datei im FS geöffnet sein
- Es darf keine Shell (oder ein anderes Programm) das aktuelle Arbeitsverzeichnis in diesem FS haben

- Konfigurationsdatei `/etc/fstab` (**file**system **table**) legt fest, welche FS beim Systemstart eingebunden werden
 - Aufbau einer Zeile der Datei:

```
# <fs>          <mount point>  <type>          <options>          <dump>  <pass>
```

- Beispiel:

```
proc           /proc          proc           defaults          0           0
/dev/sda1      /              ext3           errors=remount-ro 0           1
/dev/sda5      none           swap           sw                0           0
/dev/scd0      /media/cdrom0  udf,iso9660   user,noauto       0           0
```


- Einige Einträge haben im Optionenfeld die Option `noauto`
- Solche Einträge werden nicht automatisch gemountet, können aber einfacher von Hand gemountet werden

```
root@server:~# grep scd0 /etc/fstab
/dev/scd0 /media/cdrom udf,iso9660 user,noauto 0 0
```

```
root@server:~# mount /media/cdrom
```

- Zusatzoption `user` bedeutet: Mounten auch ohne Root-Rechte möglich

- Neben /etc/fstab gibt es noch eine Datei /etc/mtab (**mount table**)
- Diese enthält Informationen über gemountete Dateisysteme und wird automatisch (vom System) erstellt und aktualisiert

```
root@dissdevel:/# cat /etc/mtab
/dev/sda1 / ext3 rw,errors=remount-ro 0 0
tmpfs /lib/init/rw tmpfs rw,nosuid,mode=0755 0 0
proc /proc proc rw,noexec,nosuid,nodev 0 0
sysfs /sys sysfs rw,noexec,nosuid,nodev 0 0
udev /dev tmpfs rw,mode=0755 0 0
tmpfs /dev/shm tmpfs rw,nosuid,nodev 0 0
devpts /dev/pts devpts rw,noexec,nosuid,gid=5,mode=620 0 0
fusectl /sys/fs/fuse/connections fusectl rw 0 0
Daten /media/sf_Daten vboxsf gid=1001,rw 0 0
```

- Dateisysteme werden i. d. R. beim Systemstart auf Konsistenz überprüft (filesystem check)
- Auf Wunsch auch manuelle Überprüfung möglich
- Dateisystem darf dabei nicht gemountet sein
- Generisches Tool: `fsck` (**file**system **check**)

```
root@dissdevel:/# fsck /dev/sda1
fsck from util-linux-ng 2.17.2
e2fsck 1.41.12 (17-May-2010)
/dev/sda1 ist eingehängt.
```

WARNUNG!!! Die Benutzung von e2fsck auf einem eingehängten Dateisystem führt zu SCHWERWIEGENDEN SCHÄDEN im Dateisystem.

Wirklich fortfahren (j/n)?

- Automatische Überprüfung beim Systemstart:

```

Activating swap...done.
Checking root file system...fsck from util-linux-ng 2.17.2
/dev/sda1 contains a file system with errors, check forced.
/dev/sda1: 187822/623392 files (0.8% non-contiguous), 1128272/2491392 blocks
done.
Loading kernel modules...done.
Cleaning up ifupdown....
Setting up networking....
Activating lvm and md swap...done.
Checking file systems...fsck from util-linux-ng 2.17.2
done.
Mounting local filesystems...done.
  
```

- Welche Dateisysteme überprüft werden, legt letzte Spalte in `/etc/fstab` fest: 1 = prüfen

proc	/proc	proc	defaults	0	0
/dev/sda1	/	ext3	errors=remount-ro	0	1
/dev/sda5	none	swap	sw	0	0
/dev/scd0	/media/cdrom0	udf,iso9660	user,noauto	0	0

- Statt `fsck` besser direkt das für das Dateisystem passende Tool (`fsck.TYP`) aufrufen
→ dann sind auch individuelle Optionen möglich
- Beispiel `fsck.ext3`, Optionen:
 - `-f` : force, auch als „clean“ erkanntes FS prüfen
 - `-p` : versuche, Fehler automatisch zu beheben
 - `-y` : alle Fragen, die `fsck.ext3` stellt, automatisch mit „y“ (yes) beantworten
 - `-c` : Programm `badblocks` aufrufen (findet defekte Blöcke und trägt diese in Bad Blocks List ein)

- Beispiel `fsck` auf Ext3-Dateisystem

```
root@dissdevel:/# fsck /dev/sda8
fsck from util-linux-ng 2.17.2
e2fsck 1.41.12 (17-May-2010)
/dev/sda8: sauber, 11/65536 Dateien, 12644/262144 Blöcke
```

(jetzt mit `-f` erzwingen)

```
root@dissdevel:/# fsck -f /dev/sda8
fsck from util-linux-ng 2.17.2
e2fsck 1.41.12 (17-May-2010)
Durchgang 1: Prüfe Inodes, Blocks, und Größen
Durchgang 2: Prüfe Verzeichnis Struktur
Durchgang 3: Prüfe Verzeichnis Verknüpfungen
Durchgang 4: Überprüfe die Referenzzähler
Durchgang 5: Überprüfe Gruppe Zusammenfassung
/dev/sda8: 11/65536 Dateien (0.0% nicht zusammenhängend), 12644/262144
Blöcke
```

- Beispiel `fsck.ext3` – mit Fehlern

```

root@dissdevel:/home/esser# fsck.ext3 -f /dev/sda8
e2fsck 1.41.12 (17-May-2010)
Durchgang 1: Prüfe Inodes, Blocks, und Größen
Durchgang 2: Prüfe Verzeichnis Struktur
Eintrag »..« in ??? (41972) hat gelöscht/unbenutzt Inode 19152.  Bereinige<j>? ja
Eintrag »..« in ??? (42004) hat gelöscht/unbenutzt Inode 19167.  Bereinige<j>? ja
Eintrag »..« in ??? (42006) hat gelöscht/unbenutzt Inode 19167.  Bereinige<j>? ja
Durchgang 3: Prüfe Verzeichnis Verknüpfungen
Durchgang 4: Überprüfe die Referenzzähler
Durchgang 5: Überprüfe Gruppe Zusammenfassung

Die Anzahl freier Inodes ist falsch (59759, gezählt=58271).
Repariere<j>? ja

/dev/sda8: ***** DATEISYSTEM WURDE VERÄNDERT *****
/dev/sda8: 7265/65536 Dateien (0.0% nicht zusammenhängend), 44392/262144 Blöcke

```

- Die FS-spezifischen `mkfs`- und `fsck`-Tools sind meist noch unter anderen (kürzeren) Namen erreichbar:
 - `mkfs.ext3` = `mke2fs` `fsck.ext3` = `e2fsck`
 - `mkfs.ext4` = `mke2fs` `fsck.ext4` = `e2fsck`
 - `mkfs.vfat` = `mkdosfs` `fsck.vfat` = `dosfsck`
 - `mkfs.msdos` = `mkdosfs` `fsck.msdos` = `dosfsck`
- Aber: dann bei `mk*fs` aufpassen, welches das Standard-FS ist (`mke2fs`: Ext2, also nicht sinnvoll...)
- `mkfs` ohne `-t`: auch Ext2
- `vfat` und `msdos` sind identische FS

- Speicherplatz-Verbrauch
 - `df` (**d**isk **f**ree) zeigt freien Platz auf einem Datenträger (oder auf allen) an
 - `du` (**d**isk **u**sage) zeigt verwendeten Platz in einem Verzeichnis an
 - für beide Tools: mit Optionen die Ausgabe anpassen

- df

```
root@dissdevel:/tmp# df
Dateisystem      1K-Blöcke  Benutzt  Verfügbar  Ben%  Eingehängt auf
/dev/sda5        9809032   6446020   2864736    70%  /
/dev/sda1       118022124 87966344  30055780   75%  /mnt/win
tmpfs            517240    0         517240     0%  /lib/init/rw
udev            512884    176       512708     1%  /dev
tmpfs            517240    0         517240     0%  /dev/shm
```

-h = „human-readable“

```
root@dissdevel:/tmp# df -h
Dateisystem      Size  Used  Avail  Use%  Eingehängt auf
/dev/sda5        9,4G  6,2G  2,8G   70%  /
/dev/sda1       113G  84G   29G   75%  /mnt/win
tmpfs            506M    0    506M   0%  /lib/init/rw
udev            501M  176K  501M   1%  /dev
tmpfs            506M    0    506M   0%  /dev/shm
```

```
root@dissdevel:/tmp# df -h /
Dateisystem      Size  Used  Avail  Use%  Eingehängt auf
/dev/sda1        9,4G  6,2G  2,8G   70%  /
```

- du

```

esser@dissdevel:~/Daten/FOM$ du
80  ./Briefe
7300  ./BS-Alt
60324  ./BS-Praxis
20184  ./BS-Theorie/Klausur
20  ./BS-Theorie/Uebung02-Loesungen/aufgabe-b
20  ./BS-Theorie/Uebung02-Loesungen/aufgabe-c
20  ./BS-Theorie/Uebung02-Loesungen/aufgabe-d
88  ./BS-Theorie/Uebung02-Loesungen
45492  ./BS-Theorie
440  ./IT-Infrastruktur
4780  ./Material und Downloads/FOM_IT-Infrastruktur_(REP)_510-r.15_sw
31920  ./Material und Downloads
2648  ./Seminar/bearbeitet
4196  ./Seminar
149812  .
  
```

```

esser@dissdevel:~/Daten/FOM$ du -s
149812  .
  
```

← -s = summary,

```

esser@dissdevel:~/Daten/FOM$ du -sm
147  .
  
```

← -m = megabytes

- `du -s * | sort -n`

```
esser@dissdevel:~/Daten$ ls -d *
Anstel Buecher Erlangen FOM FU-Hagen Heise HM LNM privat
Promotion
```

```
esser@dissdevel:~/Daten$ du -sm * | sort -n
```

```
1  privat
3  Heise
6  Anstel
6  Buecher
9  Erlangen
15 HM
60 FU-Hagen
61 LNM
147 FOM
1715 Promotion
```

`sort -n = numerisch sortieren`

- Arbeiten am Dateisystem (für Fortgeschrittene)
- Tools für die Familie der Ext-Dateisysteme (Ext2, Ext3, Ext4)
 - `debugfs`: Eingriffe in die „Interna“ des Dateisystems
 - `dumpe2fs`: Ausgabe aller wichtigen Metadaten des Dateisystems
 - `tune2fs`: „Tuning“ für Ext-Dateisysteme, Einstellen von Optionen

debugfs, dumpe2fs, tune2fs (2)

```

root@dissdevel:/tmp# dumpe2fs /dev/sda1
dumpe2fs 1.41.12 (17-May-2010)
Filesystem volume name:   <none>
Last mounted on:         <not available>
Filesystem UUID:         27a7303b-9479-47ea-8ae8-67f9b6206920
Filesystem magic number:  0xEF53
Filesystem revision #:    1 (dynamic)
Filesystem features:     has_journal ext_attr resize_inode dir_index filetype needs_recovery
                          sparse_super large_file
Filesystem flags:        signed_directory_hash
Default mount options:   (none)
Filesystem state:        clean
Errors behavior:         Continue
Filesystem OS type:      Linux
Inode count:             623392
Block count:             2491392
Reserved block count:    124569
Free blocks:             1363120
Free inodes:             435570
First block:             0
Block size:              4096
Fragment size:          4096
Reserved GDT blocks:     608
Blocks per group:        32768
Fragments per group:    32768
Inodes per group:        8096
Inode blocks per group:  506
Filesystem created:      Tue May  3 11:55:19 2011
Last mount time:         Thu Jun  2 17:15:51 2011
Last write time:         Thu Jun  2 17:15:41 2011
Mount count:             1
Maximum mount count:     20
Last checked:            Thu Jun  2 17:15:41 2011
Check interval:          15552000 (6 months)
Next check after:        Tue Nov 29 16:15:41 2011
...

```

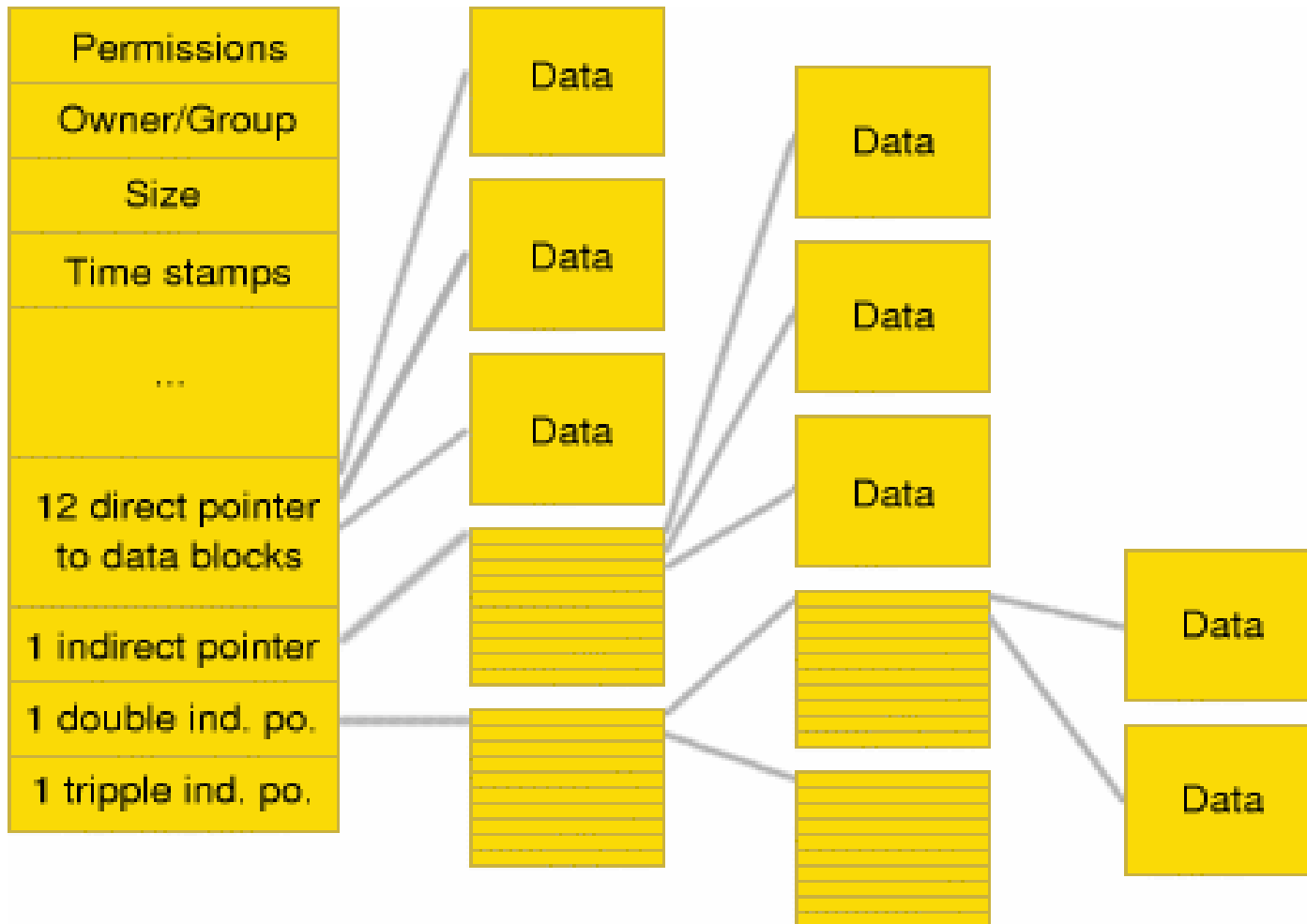
- `tune2fs`
 - Einstellen, was bei FS-Fehler passiert (continue, panic, remount-ro)
 - Intervall zwischen FS-Checks ändern
 - **Journal** ergänzen oder entfernen (→ Journaling, nächste Folie)
 - **Volume-Label** ändern
 - Größe des **reservierten Bereichs** ändern
 - dieser Teil des FS kann nur von root verwendet werden
 - für normale Nutzer erscheint das FS ggf. als voll

- Moderne Dateisysteme (z. B. ext3, ext4, ReiserFS) verwenden **Journaling**
 - Vor jeder Änderung an den **Metadaten** einer Datei wird in einen Protokollbereich (das **Journal**) die geplante Änderung geschrieben
 - Ist Änderung erfolgreich abgeschlossen, wird Eintrag aus Journal wieder gelöscht
- Beschleunigt (nach Absturz) den FS-Check:
 - nur prüfen, welche Einträge im Journal stehen – diese wurden evtl. nicht erfolgreich durchgeführt
- Variante: nicht nur Metadaten, sondern auch Daten

- Wichtige Konzepte in Linux-Dateisystemen:
 - I-Nodes
 - Dateien und Verzeichnisse
 - Datenblöcke

- I-Nodes:
 - Wenn eine neue Datei angelegt wird, sucht Linux zunächst einen freien **I-Node** (Index Node) – das ist ein Verwaltungseintrag auf der Partition
 - I-Node enthält Metadaten:
 - Dateigröße, Liste der verwendeten Blöcke
 - Besitzer und Standard-Gruppe
 - Zugriffsrechte, Timestamps ()
 - **nicht im I-Node: Dateiname und/oder Pfad (!)**
 - Danach zu I-Node Eintrag in Verzeichnis anlegen

- I-Node – grafisch:



Quelle: http://de.linwiki.org/wiki/Linuxfibel_-_System-Administration_-_Dateisysteme

- Dateien
 - Eine Datei besteht „klassisch“ aus
 - den eigentlichen Nutzdaten, die in Datenblöcken gespeichert sind,
 - einem Dateinamen (mit Pfadangabe)
 - Metadaten (Besitzer, Zugriffsrechte, Größe etc.)
 - Aus Linux-Sicht ist eine Datei zunächst die Sammlung der Datenblöcke + der I-Node (mit Metadaten und Blockliste)
 - Durch Eintragen in ein Verzeichnis (also Zuordnung: Dateiname → I-Node) wird die Datei im Dateisystem sichtbar

- Verzeichnisse

- ... sind in Linux-Dateisystemen spezielle Dateien, welche nur Zuordnungen Name → I-Node enthalten
- entspricht der Unix-Philosophie „alles ist eine Datei“
- Da Verzeichnis nur eine Datei ist, ist auch ein schnelles Verschieben eines kompletten Ordners mit Unterordnern schnell erledigt:

```
mv /home/esser/Videos /tmp/Videos
```

benötigt keine messbare Zeit (falls Verschieben innerhalb einer Partition!)

- Datenblöcke
 - Dateisystem verwaltet eine Liste freier / belegter Datenblöcke
 - Beim Löschen einer Datei werden alle verwendeten Datenblöcke als „frei“ gekennzeichnet (und bald wiederverwendet)

- Grundidee hinter Links: Datei unter mehreren Namen (und ggf. an verschiedenen Orten) ansprechen
 - **symbolische Links (soft links)**: spezielle Dateien, die den Pfad (absolut oder relativ) zu einer anderen Datei speichern
 - können „broken“ sein, also auf etwas zeigen, das es nicht gibt (wie im Web: broken link)
 - **Hard Links**: Eintrag in einem Verzeichnis, der auf denselben I-Node zeigt

- Symbolische Links / Soft Links
 - erstellen mit `ln -s` (s = soft)
 - funktionieren auch Dateisystem-übergreifend (wenn anderes FS auch eingebunden ist)

```

esser@dissdevel:~$ ls -l /mnt/windows/config.sys
-rwxr-xr-x 1 root root 36  2. Jun 20:08 /mnt/windows/config.sys
esser@dissdevel:~$ ln -s /mnt/windows/config.sys config.sys
esser@dissdevel:~$ ls -l config.sys
lrwxrwxrwx 1 esser esser 31  2. Jun 20:08 config.sys -> /mnt/windows/config.sys
esser@dissdevel:~$ ln -s /mnt/windows/BROKEN broken.txt
esser@dissdevel:~$ ls -l broken.txt
lrwxrwxrwx 1 esser esser 27  2. Jun 20:09 broken.txt -> /mnt/windows/BROKEN
esser@dissdevel:~$ cat broken.txt
cat: broken.txt: Datei oder Verzeichnis nicht gefunden
esser@dissdevel:~$ file broken.txt
broken.txt: broken symbolic link to `/mnt/windows/Windows/BROKEN'

```


- Hard Links
 - erstellen mit `ln` (ohne Option)
 - Quelle und Ziel zeigen auf gleichen I-Node
→ darum nur innerhalb eines Dateisystems möglich

```
esser@dissdevel:~$ touch datei.txt
esser@dissdevel:~$ cp datei.txt kopie.txt
esser@dissdevel:~$ ln datei.txt link.txt
```

```
esser@dissdevel:~$ ls -il *.txt
```

```
12589 -rw-r--r-- 2 esser esser 0 2. Jun 20:16 datei.txt
12590 -rw-r--r-- 1 esser esser 0 2. Jun 20:16 kopie.txt
12589 -rw-r--r-- 2 esser esser 0 2. Jun 20:16 link.txt
```

-i : I-Nodes anzeigen

rot: link
count

```
esser@dissdevel:~$ ln /mnt/windows/config.sys config.sys
```

```
ln: Erzeuge harte Verknüpfung „config.sys“ ⇒ „/mnt/windows/config.sys“:
Ungültiger Link über Gerätegrenzen hinweg
```

- Hard Links / Links / Löschen
 - Jeder Eintrag in einem Verzeichnis ist ein Link
 - Das Anlegen eines Hard Links bedeutet also nur: Für die Datei (für den I-Node!) existieren jetzt zwei Einträge in einem (oder mehreren) Verzeichnissen
 - Linux kennt intern keine „Löschen“-Operation, sondern nur eine „Unlink“-Operation
 - sie entfernt den ausgewählten Link, also die Zuordnung Dateiname → I-Node
 - und zählt den Link Count um 1 runter
 - Wenn Link Count 0 erreicht wird, wird I-Node freigegeben

Zugriffsrechte

- Jede Datei
 - ... gehört einem Benutzer (Besitzer, **user**)
 - ... und zu einer Gruppe (**group**)
- Benutzer können Mitglieder in verschiedenen Gruppen sein
- Zugriffsrechte entscheiden, ob eine Datei gelesen (**read**), geschrieben (**write**) oder ausgeführt (**execute**) werden darf

- In welchen Gruppen bin ich Mitglied?

```
$ groups
```

```
fom cdrom floppy audio dip video plugdev netdev
powerdev scanner
```

- Mitgliedschaft durch Einträge in `/etc/group` geregelt:

```
$ grep fom /etc/group
```

```
cdrom:x:24:fom
floppy:x:25:fom
audio:x:29:fom
...
fom:x:1002:fom
```

- Gruppenmitgliedschaft bearbeiten:
manuell oder (besser!) mit `gpasswd`

- Gruppe mit `gpasswd -a user group` (**add**) ergänzen:

```
# gpasswd -a fom neugr
```

 Benutzer fom wird zur Gruppe neugr hinzugefügt.

```
# groups fom
```

```
fom cdrom floppy audio dip video plugdev netdev  
powerdev scanner neugr
```
- Entfernen einer Gruppenmitgliedschaft:

```
gpasswd -d user group
```

 (**delete**)

```
# gpasswd -d fom neugr
```

 Benutzer fom wird aus der Gruppe neugr entfernt.

- Jeder Benutzer ist in einer Standardgruppe Mitglied. Welche ist das?

```
$ id
```

```
uid=1002(fom) gid=1002(fom) Gruppen=1002(fom),  
24(cdrom),25(floppy),29(audio),30(dip),...
```

- Zwei Standards für Standardgruppe
 - Debian-System: Jeder Benutzer hat seine eigene Standardgruppe (User: fom, Group: fom)
 - andere Systeme: Standardgruppe users für alle „normalen“ Benutzer
- Im Namen der Standardgruppe handeln Sie, bis Sie mit `newgrp` die Gruppe ändern.

- Neue Gruppen kann der Administrator mit `groupadd` erzeugen, um Kooperation von Teams zu erleichtern
 - z. B. mit Dateien, die für alle Gruppenmitglieder (und nur diese) les- und schreibbar sind
- Beispielszenario folgt ...

- Gruppe `profs`: Mitglieder `prof1`, `prof2`
- Gruppe `studis`: Mitglieder `anna`, `tom`, `fritz` und (!) `prof1`, `prof2`
- Ziele:
 - `profs`-Mitglieder können Daten untereinander austauschen und teilweise auch Studenten zur Verfügung stellen
 - `studis`-Mitglieder können Daten untereinander austauschen und auf die von Profs zur Verfügung gestellten Skripte, Aufgaben etc. zugreifen

- Verzeichnisstruktur

```

/srv/profs/
/srv/profs/intern/           ; Austausch der Profs untereinander
/src/profs/intern/klausuren/
/srv/profs/public/         ; Lesezugriff für Studenten möglich
/srv/profs/public/skripte/
/srv/studis/
/srv/studis/mitschriften/
/srv/studis/pruefungsprot/

```

- Gruppenzugehörigkeiten und Zugriffsrechte

- `/srv/profs/intern`: gehört Gruppe `profs`; lesen und schreiben für `profs` erlaubt, kein Zugriff für `studis`
- `/srv/profs/public`: gehört Gruppe `profs`; schreiben für `profs` erlaubt, lesen für alle (auch Nicht-Studis)
- `/srv/studis`: gehört Gruppe `studis`; lesen und schreiben für Gruppenmitglieder erlaubt

- Umsetzung: später
- Nachteil: keine vernünftige Zugriffsbeschränkung für `/srv/profs/public` möglich
→ ACLs

Unix-Dateiattribute (1)

- Neben Besitzer und Gruppe gibt es noch die sonstigen Systembenutzer (o, others)
- ergibt 9 Zugriffsrechte; Notation bei ls:

-**rwx****rwx****rwx**
 Besitzer | Grup- | sonstige
 pe

- `chown` (change owner) und `chgrp` (change group) ändern Besitzer und Gruppe einer Datei
- `chmod` (change mode) ändert Zugriffsrechte
- Beispiele:


```
chown fom /tmp/log.txt
chgrp www-data /var/www/srv1
chmod o+r /tmp/log.txt
chmod o-rwx,ug+rw /tmp/log.txt
chmod u=rw,g=r,o= /tmp/log.txt
```
- Abkürzung `a` (all) für `ogu` (`chmod a=rw ...`)

- numerische Rechte:
 - Leserecht: 4 (2^2)
 - Schreibrecht: 2 (2^1)
 - Ausführrecht: 1 (2^0)
 - aufaddieren, z. B.: rw = Lesen/Schreiben: $4+2=6$
- für Benutzer, Gruppe und Sonstige: **nnn**
 - z. B. **640**:
 - Benutzer: 6 = lesen + schreiben (nicht ausführen)
 - Gruppe: 4 = lesen (nicht schreiben, nicht ausführen)
 - Sonstige: 0 = nichts

- chmod mit numerischen Rechten nutzen
 - `rw- r-- --- = 640 (4+2+0, 4+0+0, 0+0+0)`
 - `chmod u=rw,g=r,o= /tmp/log.txt`
`chmod 640 /tmp/log.txt`
- bei der numerischen Angabe kein „Geben“ und „Nehmen“ von Rechten möglich
 (wie mit `chmod u+x ...`, `chmod o-rwx ...`)

- Beim Erzeugen einer Datei werden Standardrechte gesetzt – welche das sind, bestimmt die **UMASK (user file creation mask)**

```

$ umask                               Standard:
0022 ←                               Gruppe: nicht schreiben,
$ umask a=rw                          Sonstige: nicht schreiben
$ umask
0111
$ touch Datei; ls -l Datei
-rw-rw-rw-  1 esser users 0 2008-12-04 20:48 Datei
$ umask u=rw,g=r,o=
$ umask
0137
$ touch Test; ls -l Test
-rw-r----- 1 esser users 0 2008-12-04 20:50 Test

```


- umask wird von 666 (`rw-rw-rw-`: Standardwert für Dateien) bitweise abgezogen, um konkrete Dateirechte zu berechnen;
- Ausführrecht wird beim Erzeugen einer Datei nie vergeben

- Linux unterstützt diese klassischen Unix-Dateiattribute und einige zusätzliche...

- Dateiattribute nur auf echten Unix-Dateisystemen nutzbar – auf Windows-Datenträgern nur stark eingeschränkt:

```
# mount | grep windows
/dev/sda3 on /windows/D type vfat (rw,gid=100,umask=0002)
# touch /windows/D/Testdatei
# ls -l /windows/D/Testdatei
-rwxrwxr-x 1 root users 0 2006-12-04 21:07 /windows/D/Testdatei
# chmod a-rwx /windows/D/Testdatei
# ls -l /windows/D/Testdatei
----- 1 root users 0 2006-12-04 21:07 /windows/D/Testdatei
# umount /windows/D; mount /windows/D; ls -l /windows/D/Testdatei
-r-xr-xr-x 1 root users 0 2006-12-04 21:07 /windows/D/Testdatei
```

- Windows kennt kein Ausführattribut – wohl aber ein Read-Only-Attribut

- Bedeutung der Attribute für Verzeichnisse:
 - **read**: Verzeichnisinhalt lesen (ls in einem Verzeichnis ausführen)
 - **write**: Verzeichnisinhalt ändern (z. B. neue Datei erzeugen, Datei umbenennen)
 - **execute**: Verzeichnis betreten, also zum aktuellen Arbeitsverzeichnis machen (cd)
 - Standardrechte, von denen die umask abgezogen wird, sind bei Verzeichnissen 777 (denn x = execute steht ja für „Verzeichnis betreten“)

- Zurück zum Beispielszenario

- Ersteinrichtung:

```

root# chown -R root /srv/profs /srv/studis
root# chgrp -R profs /srv/profs/intern
root# chmod ug=rwx,o= /srv/profs/intern
root# chgrp -R profs /srv/profs/public
root# chmod ug=rwx,o=rx /srv/profs/public
root# chgrp -R studis /srv/studis
root# chmod ug=rwx,o= /srv/studis

```

- neue Dateien erzeugen:

```

prof1$ newgrp profs      # als „profs“-Mitglied arbeiten
prof1$ umask 0007       # Neue Dateien nicht für andere
prof1$ cd /srv/profs/intern
prof1$ touch pruefung.doc
prof1$ ls -l pruefung.doc
-rw-rw---- 1 prof1 profs ...      pruefung.doc

```

- Problem: Es gibt Dateien, die Benutzer nur „unter kontrollierten Bedingungen“ ändern dürfen, z. B. die Passwortdatei `/etc/shadow`
 - Änderung an der Datei mit dem Tool `passwd`
 - Dafür sind Root-Rechte nötig
 - Normale Anwender haben keine Root-Rechte
- Zwei Lösungen
 - klassisch: SUID (siehe nächste Folie)
 - neuer: `sudo` (behandeln wir hier nicht)

- Ausführbare Dateien (nur Binaries) können ein SUID- (Set User ID) und/oder ein SGID-Bit (Set Group ID) haben
 - SUID: Programm läuft immer mit den Rechten des Dateibesitzers, meist root
 - SGID: Programm läuft immer mit den Gruppenrechten der Dateigruppe (seltener verwendet)
 - Beispiel: passwd muss Systemdateien ändern

```
$ ls -l /usr/bin/passwd /etc/shadow
-rwSr-xr-x 1 root root    ... /usr/bin/passwd
-rw-r----- 1 root shadow  ... /etc/shadow
```

- SUID- und SGID-Bits mit chmod setzen

```
# cp /usr/bin/passwd /tmp/mypasswd
# chmod u-s,g+s /tmp/mypasswd
# ls -l /usr/bin/passwd /tmp/mypasswd
-rwSr-xr-x 1 root root    ... /usr/bin/passwd
-rwxr-Sr-x 1 root shadow  ... /etc/shadow
```

- s-Bits erscheinen in der ls-Ausgabe immer an der Stelle, wo sonst das x steht
- Diese Bits sind bei Shell-Skripten wirkungslos (in einigen älteren Unix-Versionen funktionierte das auch mit Skripten)

- Ext2-/Ext3-Extra-Flags (immutable, append-only etc.) mit `chattr` bearbeiten

NAME

chattr - change file attributes on a Linux second extended file system

SYNOPSIS

chattr [-RV] [-v version] [mode] files...

DESCRIPTION

chattr changes the file attributes on a Linux second extended file system.

The format of a symbolic mode is `+--[ASacDdIijsTtu]`.

The operator ``+'` causes the selected attributes to be added to the existing attributes of the files; ``-'` causes them to be removed; and ``='` causes them to be the only attributes that the files have.

- Beispiel für `chattr`:

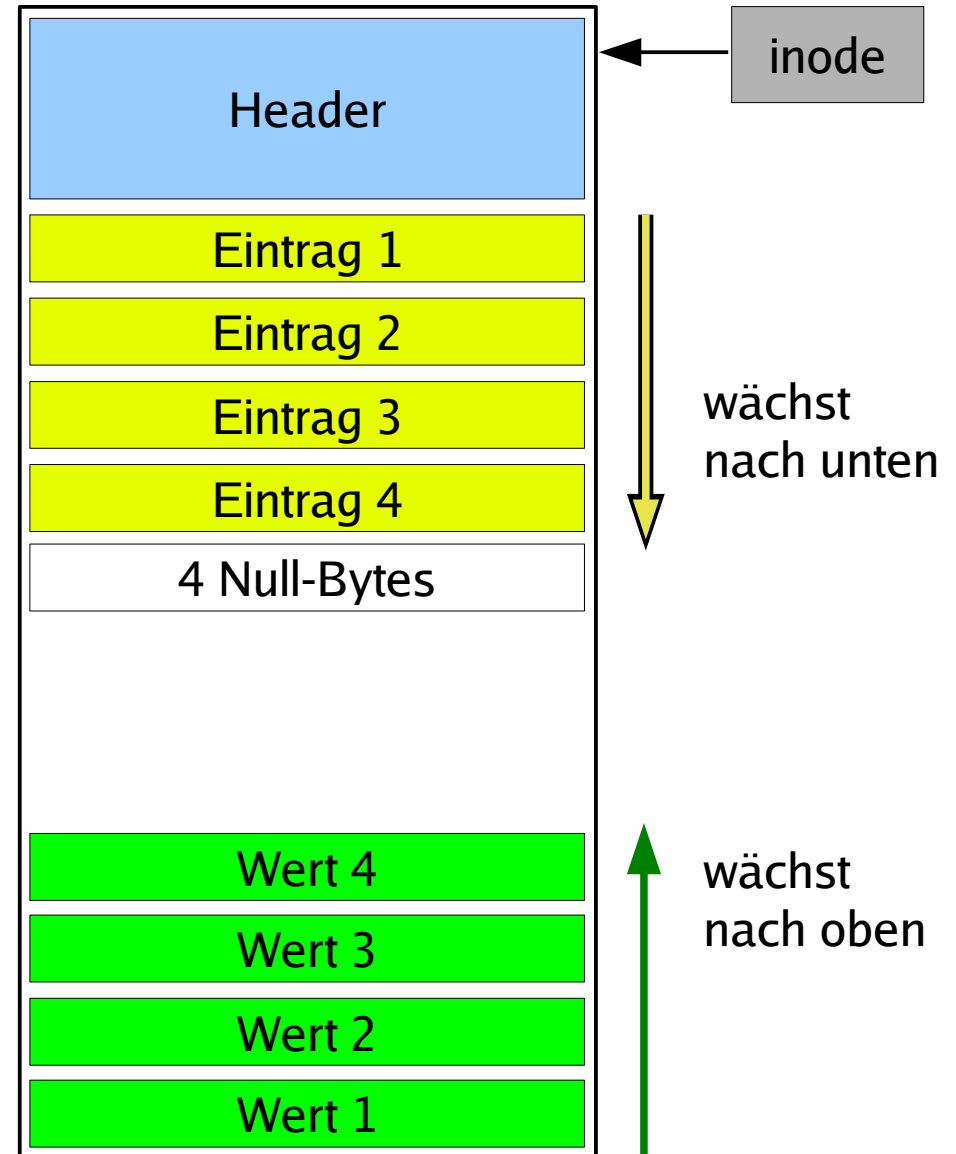
```
# cd /tmp; touch logdatei
# chattr +a logdatei
# echo Hallo >> logdatei           # >> = anhängen
# echo Welt >> logdatei
# cat logdatei
Hallo
Welt
# echo Ueberschreiben > logdatei
bash: logdatei: Die Operation ist nicht erlaubt
```

- Attribute anzeigen mit `lsattr`:

```
# lsattr -l logdatei
logdatei                Append_Only
```

Erweiterte Attribute (1)

- Erweiterte Attribute speichern beliebige Name-/Wert-Paare, u. a. ACLs
- Inode-Größe: 128 Byte
 - kein Platz für erweiterte Attribute
 - Vergrößerung auf 256 Byte nicht effizient
- Lösung: Separater Block für extended attributes



- bearbeiten mit `setfattr`, `getfattr`, `attr`:

```
amd64:/home/esser # setfattr -n user.foo -v betriebssysteme test.txt
amd64:/home/esser # getfattr -d test.txt
# file: test.txt
user.foo="betriebssysteme"
```

```
amd64:/home/esser # attr -g user.foo test.txt
Attribute "user.foo" had a 15 byte value for test.txt:
betriebssysteme
```

- Software ist auf dem Debian-System nicht installiert → `apt-get install attr`
- Verwaltung von ACLs über das Paket `acl` → `apt-get install acl`
 - Tools: `getfacl`, `setfacl`

Nicht verwechseln:

- Standard-Unix-Dateiattribute
 - UID, GID
 - Standardzugriffsrechte rwx für user/group/others
 - Zugriffszeiten, ...
- Extra-Flags
 - immutable, compressed, secure deletion, ...
- Extended Attributes
 - beliebige, frei definierbare Attribute (inkl. ACLs)