Linux Grundlagen / Die Kommandozeile

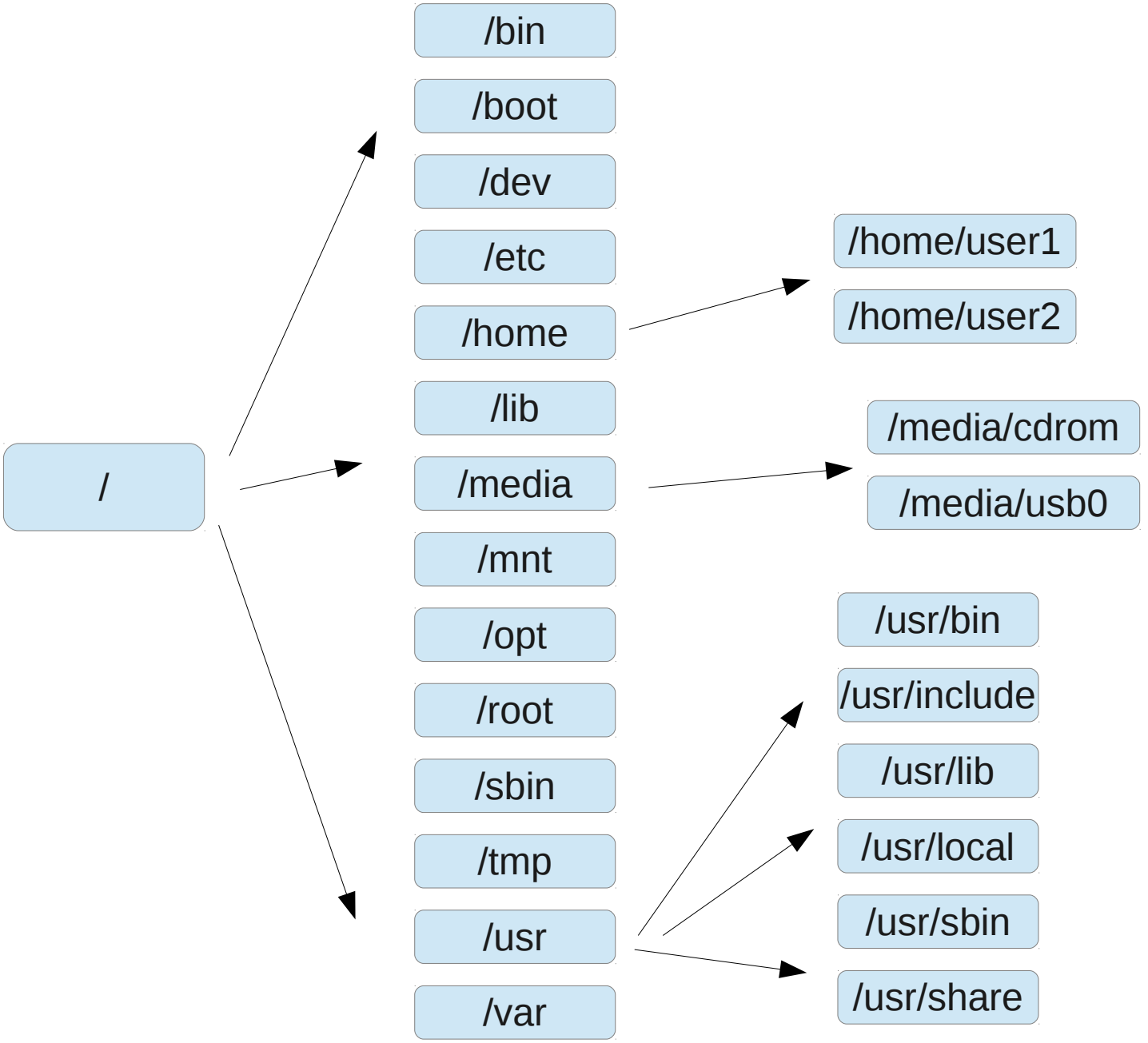
Teil 2: Das Linux Filesystem

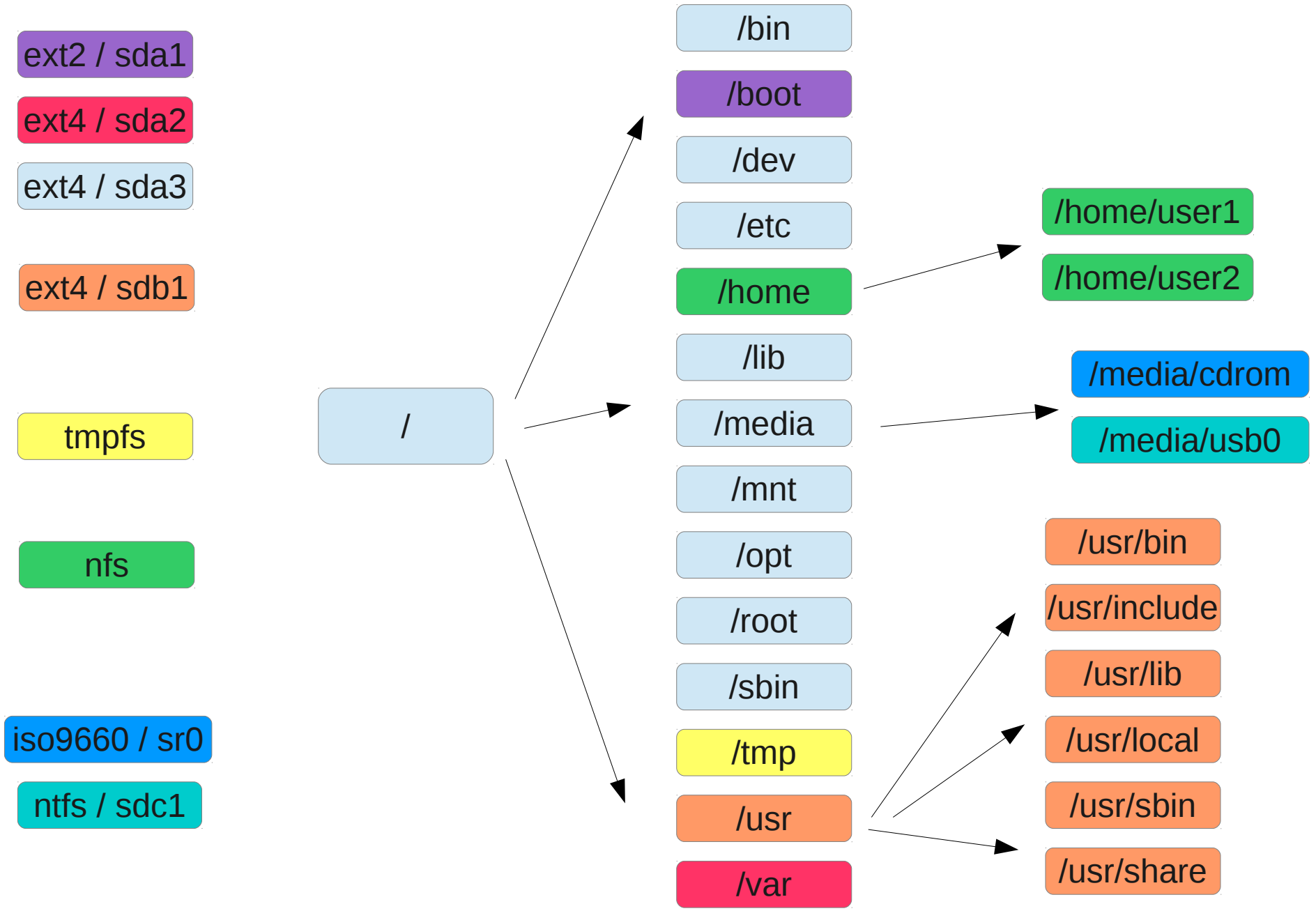
Häufig die erste Frage eines Linux-Einsteigers:

Wo ist denn das C: Laufwerk?

Linux ist ein Unix(artiges) System, und da ist alles etwas anders...

1969:	Filesystem eines der allerersten Komponenten des "Ur-UNIX" (Bell Labs, Thompson / Ritchie an einer PDP-7) Einführung von `/bin /dev /etc /lib /tmp /usr`
Ende 70er:	Abtrennung von `/sbin` aus `/etc`
80er:	Abtrennung variabler Verzeichnisse aus `/usr` in `/var`
1984:	Einführung von /proc in Unix V8
1993:	Entwicklung des Filesystem Hierarchy Standard (FHS) Zunächst nur auf Linux bezogen
1995:	BSD-Entwickler springen auf die FHS Entwicklung auf. (Ziel: Gemeinsamer UNIX-Standard)
2003:	Kernel 2.5 führt sysfs ein (gemountet auf /sys)
2004:	Version 2.3 des FHS wird veröffentlicht (aktuell gültige Version)
2011:	Fedora führt `/run` ein. Diskussion über Vereinigung von `/bin` und `/sbin` mit `/usr`


/
/bin
/boot
/dev
/etc
/home
/lib
/media
/mnt
/opt
/root
/sbin
/tmp
/usr
/var
/home/user1
/home/user2
/media/cdrom
/media/usb0
/usr/bin
/usr/include
/usr/lib
/usr/local
/usr/sbin
/usr/share


ext2 / sda1
ext4 / sda2
ext4 / sda3
ext4 / sdb1
tmpfs
nfs
iso9660 / sr0
ntfs / sdc1
/
/bin
/boot
/dev
/etc
/home
/lib
/media
/mnt
/opt
/root
/sbin
/tmp
/usr
/var
/home/user1
/home/user2
/media/cdrom
/media/usb0
/usr/bin
/usr/include
/usr/lib
/usr/local
/usr/sbin
/usr/share

mount bindet Filesysteme in den Verzeichnisbaum ein:

mount <device> <dir>

```
$> mount /dev/sdc1 /mnt
```

Optionen:

```
-t <type>          # Explizite Angabe des Filesystem-Typ
-o <options>       # Filesystem-spezifische Optionen
-a                 # Mounte alle Einträge in der fstab
```

`mount` **`-o <options>`** `<device> <dir>`

`loop`	Verwende ein Loop-Device (Einbinden von Files als Gerät)
`ro`	FS als *Read-Only* einbinden
`remount`	Optionen eines schon eingebundenen FS ändern
`noauto`	FS wird nicht bei `mount -a` mit eingebunden
`user`	Jeder Benutzer kann das FS in den Verzeichnisbaum einbinden
`users`	Wie *`user`*, aber auch ein anderer Benutzer kann das FS unmounten

Steuern der Zugriffsrechte bei Filesystemen ohne UNIX-Style permissions:

`uid=<value>`	Setze den Besitzer des FS
`gid=<value>`	Setze die Gruppe des FS
`umask=<value>`	Setze die umask des FS

mount ohne Optionen zeigt die aktuell eingehängten Dateisysteme an:

```
$> mount
/dev/sda3 on / type ext4 (rw,noatime,errors=remount-ro)
/dev/sda1 on /boot type ext4 (rw,noatime)
/dev/sda2 on /var type ext4 (rw,noatime)
/dev/sda6 on /home type ext4 (rw,noatime)
[...]
```

umount hängt eingebundene Filesysteme wieder aus:

umount <device>|<dir>

```
$> umount /mnt
```

Achtung: Filesysteme mit noch offenen Dateien können nicht ausgehängt werden!

```
~$> mount /dev/sdc1 /media/usb0
~$> cd /media/usb0
/media/usb0$> umount /media/usb0
umount: /media/usb0: device is busy.
        (In some cases useful info about processes that use
         the device is found by lsof(8) or fuser(1))
/media/usb0$> cd ~
~$> umount /media/usb0
~$> _
```

`/etc/fstab` enthält Informationen zu den lokalen Filesystemen:

`<device>`	Das einzuhängende Gerät
`<mount pt>`	Einhängepunkt
`<type>`	Typ des Filesystems (unterstützte in /proc/filesystems)
`<options>`	Einhängeoptionen
`<dump>`	Steuert Verhalten des Backuptools *dump* (veraltet)
`<pass>`	Reihenfolge bei Überprüfung von fsck (0 = kein Check)

```
<device>   <mount pt> <type> <options>           <dump><pass>

/dev/sda6  /home       ext4  noatime,data=writeback 0    2

/dev/sdb1  /media/usb0 auto  rw,user,noauto         0    0

tmpfs      /tmp        tmpfs mode=1777             0    0
```

Loop-Devices bieten die Möglichkeit, Filesystem-Image-Dateien einzubinden.

Problem: Linux kann nur Block-Devices einbinden.

Lösung: Umweg über ein Loop-Device:

```
$> losetup /dev/loop0 fsimage.img
$> mount /dev/loop0 /mnt
$> [...]
$> umount /mnt
$> losetup -d /dev/loop0
```

mount kann Einbinden per Loop-Device auch per Option durchführen:

```
$> mount -o loop fsimage.img /mnt
$> [...]
$> umount /mnt
```

- **Static** Nur von 'root' zu ändern (also meistens selten)
- **Variable** Von allgemeinen Aktionen zu ändern (also meistens häufig)

- **Shareable** Von mehreren Hosts teilbar (-> Netzwerk)
- **Unshareable** Für einen Host spezifisch (-> auf lokalem Datenträger)

	Shareable	Unshareable
Static	/usr /opt	/etc /boot
Variable	/home	/var/run /var/lock

Achtung:

Der FHS ist eher eine "Richtlinie" denn ein fester Standard.

Viele Dinge im FHS sind historisch gewachsen und bei modernen Systemen mitunter nicht mehr sinnvoll.

Viele Linux-Distributionen weichen vom FHS ab und ergänzen das Filesystem.

=> siehe z.B. Diskussion um Vereinigung von `/bin` und `/sbin` mit `/usr`

=> *GoboLinux* mit radikal anderer Verzeichnisstruktur

Manpage zur Filesystem-Hierarchie:

```
> man hier
```

/home

Shareable Variable

Benutzerverzeichnisse, organisiert in `/home/<username>`

Bei sehr großen Organisationen auch `/home/<division>/<username>`

Nur wenig standardisiert; Benutzer darf frei mit seinem Home-Verzeichnis arbeiten

Ablageort für Benutzer-eigene Konfigurationsdateien; verborgen mit `.<conffile>`

Vorlage für neue Benutzerverzeichnisse in `/etc/skel`

`/home/<user>/.bashrc`	Startupbefehle für eine "interactive non-login shell"
`/home/<user>/.profile`	Startupbefehle für eine "interactive login shell"

/mnt **/media**

`/media` Mountpunkt für **Wechselmedien**

`/media/cdrom`
`/media/usb0`
`/media/floppy`

`/mnt` **Temporärer** Mountpunkt (meist von root genutzt)

Tipp: Verzeichnisse in `/media` mit Option **`user`** in die fstab eintragen.

/

Unshareable
Static

Wurzelverzeichnis

Soll außer den Verzeichnissen des FHS keine weiteren Dateien enthalten.

Ausnahme: Kernel und Initrd (kann auch Symlink nach `/boot` sein)

`/vmlinuz`	Kernel
`/initrd.img`	Initrd ('Initial Ramdisk')

/root

Shareable
Static

Home-Verzeichnis für den **Superuser**

Sollte **sparsam** und nur für Zwecke der **Systemadministration** verwendet werden!

/bin **/sbin**

Unshareable
Static

Grundlegende Systemkommandos, die vorhanden sein müssen wenn keine anderen Filesysteme gemountet sind.

`/bin` Grundlegende Binaries, verwendet von **allen Usern**

Enthält Programme wie
`ls, echo, sh, mkdir, cp, cat, ps, rm, mount, ...`

`/sbin` Grundlegende Binaries für die Systemadministration (**nur root**)

Enthält u.a.
`fdisk, fsck, mkfs, mkswap, swapon, shutdown, init, ...`

/lib

Unshareable
Static

Shared libraries, die vorhanden sein müssen wenn kein anderes Filesystem gemountet ist. (von Binaries in `/bin` und `/sbin` benötigt).

`/lib64/`	64-bit Libraries
`/lib/modules/<version>`	Kernelmodule
`/lib/firmware/`	Firmware für Hardwarekomponenten

/usr

Shareable
Static

Unterhierarchie, **shareable** mit anderen Hosts (also z.B. auslagerbar auf Netzwerk)

`/usr/bin`	Die meisten User-Programme
`/usr/include`	C-Header Files
`/usr/lib`	Libraries (shared and static)
`/usr/local`	Weitere Unterhierarchie für manuell installierte Pakete
`/usr/sbin`	Systemadministrationskommandos
`/usr/share`	Daten unabhängig von der Systemarchitektur (man-Pages, Dokumentation, Icons etc.)
`/usr/src`	Source code (z.B. Kernel-Sourcen)

/opt

Shareable
Static

Verzeichnis für **Addon**-Softwarepakete

Ursprünglich: Verzeichnis für Drittanbieter-Pakete

Pakete sollten folgendermaßen installiert werden:

```
/opt/<package>/          # per Paketname
/opt/<provider>/         # per Herstellername
```

/srv

Shareable
Static

Daten, die von System als Server angeboten werden.

Achtung: Konkurriert je nach Distribution mit `/var`
(Webserver mitunter unter `/var/www`)

`/srv/www`	Webseiten
`/srv/ftp`	FTP-Verzeichnis
`/srv/samba`	Samba-Share

/var

Mixed Variable

Zweck: Trennung **variabler** Daten von /usr.
Enthält Logging-Daten, Print / Mail-Spool, Caches, etc.

`/var/cache`	Cachedaten
`/var/lib`	Statusdaten
`/var/spool`	Spooldaten (Printjobs etc.)
`/var/tmp`	Temporäre Daten, bleibt bei Reboot erhalten
`/var/log`	Logfiles
`/var/mail`	Mailboxen
`/var/local`	Variable Daten für `/usr/local`
`/var/opt`	Variable Daten für `/opt`
`/var/lock`	Lockfiles
`/var/run`	Runtime data

← Unshareable!

/run

Unshareable Variable

Nicht im FHS 2.3,
aber inzwischen bei vielen Distributionen vorhanden, sowie im FHS 3.0 beta

Auslagerung der *unshareable* Verzeichnisse aus `/var`

Enthält lokale Run-Time Daten, deshalb häufig als **tmpfs** gemountet (Ram-Disk).

`/run`	Verlinkt von /var/run
`/run/lock`	Verlinkt von /var/lock

/tmp

Shareable Variable

Speicherort für **temporäre** Dateien.

Programme sollten nicht annehmen, daß in `/tmp` gespeicherte Dateien nach Neustart noch vorhanden sind.

Häufig wird `/tmp` als **`tmpfs`** gemountet.

`/tmp` hat Berechtigungen `rwxrwxrwt / 1777`

Alle Benutzer haben Lese / Schreib / Ausführungsrechte für `/tmp`.

`/tmp` ist das Lehrbuchbeispiel für Verwendung des **Sticky Bit.**

/etc

Unshareable
Static

`/etc/passwd`	Benutzerkonfiguration
`/etc/group`	Gruppen Konfiguration
`/etc/shadow`	Passwortdatei
`/etc/fstab`	statische Konfiguration der Dateisysteme
`/etc/sysctl.conf`	Konfiguration Kernelparameter
`/etc/hosts`	Statische Tabelle Auflösung Rechnernamen
`/etc/inittab`	Init Konfiguration (System V Init)
`/etc/init.d/`	Kontrollskripte für Systemdienste
`/etc/rc<X>.d/`	Start/Stop Skripte für Runlevels
`/etc/crontab`	Cron-Job Tabelle (regelmäßige Ausführung)
`/etc/cron.<X>/`	Cron-Skripte
`/etc/X11/`	Konfiguration des X-Window Systems

/boot

Unshareable
Static

Dateien notwendig für den **Bootvorgang:**

`/boot/vmlinuz-<version>`	Linux-Kernel
`/boot/initrd.img-<version>`	Boot-Ramdisk
`/boot/System.map-<version>`	Kernel Einsprungtabelle
`/boot/config-<version>`	Kernel Konfiguration
`/boot/grub/`	Konfiguration des grub-Bootloaders

/dev

Unshareable Variable

Veraltet: Device-Dateien per `mknod` oder `devfs`
Aktuell: `/dev` wird durch `udev`-Daemon gefüllt

`/dev/hd<X><P>`	Partition auf IDE-Laufwerk
`/dev/sd<X><P>`	Partition auf SCSI-Laufwerk (auch SATA oder USB)
`/dev/loop<X>`	Loop-Devices

`/dev/zero`	Datensenke	(Lesen: '0'	Schreiben: Verwerfen)
`/dev/null`	'Leeres' File	(Lesen: EOF	Schreiben: Verwerfen)
`/dev/full`	'Volles' Gerät	(Lesen: '0'	Schreiben: ENOSPC)

`/dev/random`	Echte Zufallszahlen (blockiert wenn Entropiepool verbraucht)
`/dev/urandom`	Pseudo-Zufallszahlen ('unblocking random')

/proc

Unshareable Variable

Pseudofilesystem, Schnittstelle zum Kernel (`procfs`)

`/proc/<pid>`	Information zu Prozess <pid>
`/proc/cpuinfo`	Informationen zur CPU
`/proc/meminfo`	Informationen zum Arbeitsspeicher
`/proc/swaps`	Informationen zum Swap-Speicher
`/proc/sys`	Systemparameter (=> `/etc/sysctl.conf`)
`/proc/sys/kernel`	Kernel Parameter
`/proc/sys/net`	Netzwerk-Subsystem Parameter
`/proc/sys/vm`	Speicherverwaltung Parameter

/sys

Unshareable Variable

Pseudofilesystem, Informationen über Kernelsubsysteme und Geräte. (`sysfs`)

Jedes Kernel Objekt (kobject) erhält einen Eintrag in `/sys`

Ab Kernel 2.5, Zweck: Entlastung von `/proc`

Beispiele:

```
/sys/block/sda/queue/scheduler      # I/O Scheduler der 1. Festplatte
/sys/devices/platform/smapi/BAT0/   # Batterieinformationen bei Thinkpads
```

lost+found

Special

Vorhanden im der Wurzelebene jedes einzelnen Dateisystems

lost+found ist nur bei bestimmten Dateisystemtypen vorhanden, insbesondere bei den **EXT**-Dateitypen.

Zweck: **"Fundbüro"** für Dateifragmente, die beim Check & Reparatur eines fehlerhaften Dateisystems per fsck gefunden werden.