

1 Systemadministration

- grundlegende Aufgaben eines Systemverwalters
 - System installieren und konfigurieren
 - Netzwerk konfigurieren
 - *Updates* installieren und ggf. System anpassen
 - zusätzliche Hardware und Software installieren
 - Fehlerbehebung

- Routineaufgaben eines Systemverwalters
 - Benutzer verwalten und beraten
 - Datensicherung (*Backup / Restore*)
 - Netzwerkmanagement
 - Systemsicherheit und Überwachung

- Empfehlungen
 - System- und Anwender-Software in verschiedenen Bereichen der Festplatte installieren
(einfacheres *Update* der System-Software; erleichtert eine spätere Neuinstallation)
 - Anwender-Software sollte möglichst ohne *Root*-Privilegien installiert werden
 - System-Software sollte möglichst unverändert installiert werden
⇒ jede Änderung **muß** notiert werden
 - ggf. zu jeder Maschine ein Buch mit folgenden Angaben führen
 - ◆ Liste aller Veränderungen
 - ◆ aufgetretene Probleme und deren Lösung
 - Strategie für die Konfiguration der Systeme und Vergabe der Privilegien festlegen
⇒ erleichtert gemeinsame Nutzung der Ressourcen
⇒ erlaubt effiziente Verwaltung der Systeme

1.1 Installation von Betriebssystemen

- Grundprinzip

1. Die Maschine befindet sich im Konsolen-Modus (nicht bei PC's)
2. über einen speziellen Befehl wird das Gerät angesprochen, auf dem sich das zu installierende Betriebssystem befindet
(z.B. bei *Sun Sparc*: *b cdrom* oder *boot cdrom*, bei PC's wird das Medium in das entsprechende Laufwerk gelegt und <Ctrl><Alt> bzw. <Strg><Alt><Entf> oder die *Reset*-Taste gedrückt)
3. es erscheint ein Installationsmenü, das durch die weitere Installation führt
(i.a. steht jetzt ein kleines Betriebssystem auf dem Datenträger bzw. in der *RAM-Disk* zur Verfügung oder es wurde eine *Miniroot* im *Swap*-Bereich der Festplatte erzeugt oder ... ; mit diesem Mini-Betriebssystem kann dann die Partitionierung und Formatierung der Festplatte vorgenommen werden)
4. i.a. kann jetzt zwischen einer Standard-Installation und einer Benutzer-definierten Installation gewählt werden
5. im Laufe der Installation müssen i.a. viele Fragen beantwortet werden
(Landessprache, Zeitzone, u.U. Fragen zu den Hardware-Eigenschaften, Name des Rechners, IP-Adresse, ...)

- *Update* oder Neuinstallation

- *Update*: Einspielen einer neuen Version in eine vorhandene Installation

- ◆ die Partitionierung der Festplatte bleibt erhalten
- ◆ die Konfiguration des Systems bleibt i.a. erhalten
- ◆ zusätzliche Software bleibt i.a. lauffähig (ggf. *Update* erforderlich)

⇒ nicht möglich, wenn das neue Betriebssystem größere Partitionen benötigt

⇒ **vorher** unbedingt alle Daten sichern!

⇒ i.a. weniger Nacharbeit, da lokale Konfigurationen erhalten bleiben

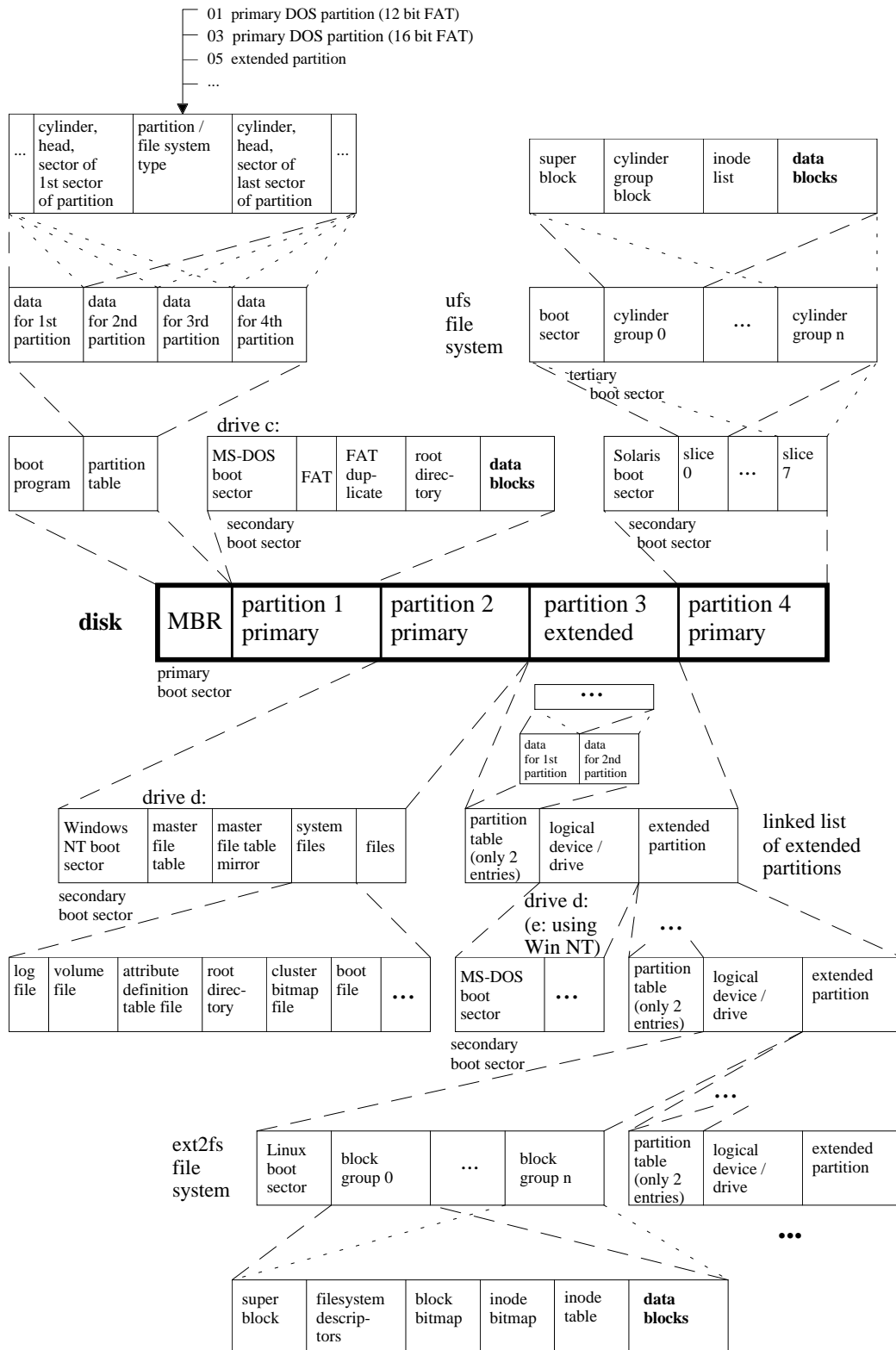
- Neuinstallation

- ◆ die Partitionierung der Festplatte muß durchgeführt werden
- ◆ das System muß i.a. konfiguriert werden
- ◆ zusätzliche Software muß eingespielt und konfiguriert werden

⇒ u.U. schneller, wenn nicht jedes *Update* eingespielt wurde

⇒ u.U. problemloser, wenn Systemdateien verändert wurden (in diesen Fällen führt ein *Update* u.U. nicht zum Erfolg, da es dann manchmal nicht korrekt durchgeführt werden kann)

- Beispiel: Layout einer Festplatte in einem PC



- Besonderheiten bei PC's
 - bis zu vier Partitionen je Festplatte in der Partitionstabelle
 - höchstens eine erweiterte Partition
 - jede Partition (heißt bei Solaris logische Platte, die ihrerseits in *slices* eingeteilt wird) kann für unterschiedliche Dateisysteme formatiert werden
 - jede Partition, von der ein Betriebssystem gestartet werden soll, muß bei älteren Betriebssystemen in den ersten 1024 Zylindern der Festplatte beginnen
(aus diesem Grund liefert eine *low-level*-Formatierung einer 9 GB großen Festplatte z.B. eine logische Platte mit 1115 Zylindern à 255 Köpfen (Oberflächen) à 63 Sektoren à 512 Bytes, so daß ca. die ersten 8GB der Festplatte für solche Partitionen zur Verfügung stehen)
 - die meisten (alle?) Betriebssysteme können nur von der ersten Festplatte gestartet werden
 - manche Betriebssysteme können sowohl von primären Partitionen als auch von logischen Laufwerken der erweiterten Partition gestartet werden
 - manche Betriebssysteme können nur von der ersten primären Partition des ersten Laufwerks gestartet werden

- Probleme bei mehreren Betriebssystemen auf einem Rechner (PC)
 - Auswahl der Hardware
(nicht jedes Betriebssystem hat für jeden *Controller* einen Treiber)
 - manche Betriebssysteme manipulieren die Partitionstabelle, so daß sich andere Betriebssysteme danach nicht in der gewünschten Form installieren lassen
 - ⇒ die Installationsreihenfolge ist u.U. wichtig
(Windows NT muß z.B. vor Solaris 7 x86 installiert werden, da Solaris 7 die Solaris-Partition an Position 1 der Partitionstabelle verschiebt und NT dort Laufwerk *c:* erwartet)
 - die Partitionierungsprogramme der einzelnen Systeme mögen sich manchmal nicht
(ein Betriebssystem sieht seine Partition nicht; am besten jede Partition mit dem Programm des jeweiligen Betriebssystems anlegen)
 - die Numerierung der Partitionen durch das Partitionierungsprogramm und den Betriebssystemkern unterscheiden sich
(spezielles Problem von Linux 2.2.x; wenn Linux vollständig in einer erweiterten Partition installiert wird und Solaris x86 in einer primären Partition, die vor der erweiterten Partition liegt, kann Linux nicht mehr gestartet werden, da es die *Solaris-Slices* wie Laufwerke einer erweiterten Partition behandelt und der Linux-Kern diese in diesem Fall vor die Partitionen der erweiterten Partition anordnet)
 - ⇒ die Anordnung der Systeme auf der Festplatte ist u.U. wichtig
 - Die Dateisystemkennungen sind nicht eindeutig
(die Kennung 82 wird sowohl zur Kennzeichnung einer *Swap*-Partition von Linux als auch zur Kennzeichnung einer Solaris-Partition benutzt)
 - ⇒ die Systeme können sich u.U. gegenseitig zerstören

- mögliche Zuordnung der Betriebssysteme zu den Partitionen
 - 1. primäre Partition MS-DOS / Windows
 - 2. primäre Partition Windows NT / 2000
 - 3. erweiterte Partition Linux
 - 4. primäre Partition Solaris x86

- mögliche Installationsreihenfolge
 - 1. MS-DOS
 - 2. Linux
 - 3. Windows NT / 2000
 - 4. Solaris x86

- falls MS-DOS nicht installiert wird, sollte trotzdem eine FAT16-Partition vorhanden sein, da über diese Partition Daten zwischen den verschiedenen Betriebssystemen ausgetauscht werden können

- die Solaris-Partition muß als aktive Partition gekennzeichnet sein (sonst kann Solaris nicht gestartet werden)

- nach jeder erfolgreichen Installation sollte der MBR gerettet werden, damit man nach einer fehlerhaften Installation wieder einen Aufsetzpunkt hat

Beispiel für Linux:

```
mkdir /msdos  
mount -t msdos /dev/sda1 /msdos    oder    mount -t msdos /dev/hda1 /msdos
```

retten:

```
dd if=/dev/sda of=/msdos/mbr.{dos, lin, sol, nt} bs=512 count=1    oder  
dd if=/dev/hda of=/msdos/mbr.{dos, lin, sol, nt} bs=512 count=1
```

wiederherstellen:

```
dd if=/msdos/mbr.{dos, lin, sol, nt} of=/dev/sda bs=512 count=1    oder  
dd if=/msdos/mbr.{dos, lin, sol, nt} of=/dev/hda bs=512 count=1
```

Beispiel für Solaris x86:

```
mkdir /msdos  
mount -F pcfs /dev/dsk/c0t0d0p0:1 /msdos    oder  
mount -F pcfs /dev/dsk/c0d0p0:1 /msdos  
dd if=/dev/dsk/c0t0d0p0 of=/msdos/mbr.{dos, lin, sol, nt} bs=512 count=1  
oder  
dd if=/dev/dsk/c0d0p0 of=/msdos/mbr.{dos, lin, sol, nt} bs=512 count=1
```

- Partitionsnamen

- 1) MS-DOS, Windows 3.11 / 95 / 98 / NT / 2000, OS/2

Laufwerksbuchstaben, z.B. c:, d:, e:, ...

- 2) Linux

EIDE: /dev/hd{Gerätebuchstabe}[/{Partitionsnummer}]

SCSI: /dev/sd{Gerätebuchstabe}[/{Partitionsnummer}]

(die erste Festplatte erhält den Buchstaben *a*, die zweite *b*, usw.; die Partitionsnummern 1 bis 4 bezeichnen die Partitionen im MBR und 5 bis 15 logische Laufwerke in der erweiterten Partition; falls die Partitionsnummer fehlt, ist die gesamte Festplatte gemeint)

- 3) Solaris

EIDE: /dev/dsk/cwdx{*sa*, *pb*} block device

 /dev/rdisk/cwdx{*sa*, *pb*} raw device

w logical controller number
(immer 0, wenn es nur einen *Controller* gibt)

x drive number

a slice number (0-7) oder

b partition number (0-4)

SCSI: /dev/dsk/cwtxdy{*sa*, *pb*} block device

 /dev/rdisk/cwtxdy{*sa*, *pb*} raw device

w logical controller number
(immer 0, wenn es nur einen *Controller* gibt)

x SCSI-bus target number (0-6)

y drive number

(*logical unit number* (LUN) des Laufwerks, falls das Gerät mehrere logische Einheiten unterstützt; i.a. immer 0)

a slice number (0-7) oder

b partition number (0-4)

- *slice 2* bezeichnet die gesamte logische Platte
- Auswahl des *block* oder *raw device* ist abhängig vom Kommando

Aufgabe 1-1:

Installieren Sie auf Ihrer Festplatte die Betriebssysteme Linux und Solaris x86. Sie können über das Netz vom *Server* "swlab92" (193.174.24.92) installiert werden. Die SuSE-Distribution befindet sich im Verzeichnis /export/SuSE/8.1 und Solaris im Verzeichnis /export/install/x86_2.8. Sie sollten Ihre Festplatte folgendermaßen partitionieren:

Partition	Typ	Größe	System	wofür?
1	primär	512 MB	MS-DOS	Benutzer
2	extended	5.120 MB		Linux
3	primär	5.120 MB		Solaris
4	primär	Rest		Windows 2000
	log. Laufwerk	200 MB	ext2	Linux /
	log. Laufwerk	120 MB	swap	Linux swap
	log. Laufwerk	200 MB	ext2	Linux /var
	log. Laufwerk	2.048 MB	ext2	Linux /opt
	log. Laufwerk	2.048 MB	ext2	Linux /usr
	log. Laufwerk	Rest	ext2	Linux /export/home

Für Solaris sollten folgende *Slices* angelegt werden:

Slice	Größe	wofür?
0	200 MB	/
1	120 MB	swap
2	vorgegeben	gesamte logische Platte (overlap)
3	200 MB	/var
4	-	-
5	2.048 MB	/opt
6	2.048 MB	/usr
7	Rest	/export/home

- Netzwerk-Konfiguration der Rechner

Rechner	IP-Adresse
swlab71	193.174.24.71
swlab72	193.174.24.72
...	...
swlab91	193.174.24.91
swlab92	193.174.24.92 (<i>Server</i>)

Netzmaske: 255.255.255.224

Domäne: informatik.fh-fulda.de

- Hardware-Konfiguration der Rechner

	Name	horiz. Frequenz	vert. Frequenz
17" Monitor	Belinea 10 30 80	30-96 kHz	50-160 Hz
Graphikkarte	GeForce 2 MX, 32 MB	-	-
Netzwerkkarte	3COM 3C905C-TX	-	-

Controller	IRQ	I/O
Graphikkarte	5	-
Netzwerkkarte	11	0xC000

Microsoft IntelliMouse am PS/2-Anschluß

CDROM / DVD: Toshiba DVD-ROM SD-M1612

- Vorgehensweise

1. Rechner mit MS-DOS-*Boot*-Diskette starten und MS-DOS-Partition anlegen

("fdisk", "format /s c:", "xcopy /e /v /-y a: c:", Konfigurationsdateien "config.sys", "autoexec.bat" und "c:\cmd\norenv.bat" anpassen, so daß MS-DOS von Festplatte gestartet werden kann)

2. Linux-CD einlegen und Rechner erneut starten

- a) Installation "Safe settings" und Bildschirmauflösung "1024x768" auswählen, Installation starten und alles bestätigen

- b) deutsche Sprache und deutsche Tastatur auswählen

- c) Netzwerk-Modul laden

- *Kernel-Module (Hardware-Treiber)* wählen

- Netzwerk-Treiber *3c90x* wählen

- zurück in das Hauptmenü

- d) *Installation / System starten* wählen

Installation / Update starten wählen

Installationsart *Netzwerk (NFS)* wählen

- e) kein DHCP

- f) IP-Adresse und Netzmaske des lokalen Rechners benutzen

- g) *Default-Gateway* 193.174.24.65, *Nameserver* 193.174.25.38 oder 193.174.26.133 und *NFS-Server swlab92* benutzen

- h) Verzeichnis der Linux-Distribution eingeben

(Installationsroutinen vom *NFS-Server* werden geladen und gestartet; CD kann aus dem Laufwerk genommen und weitergegeben werden)

- i) "Modul: usb-storage", "Modul: lvm-mod" und "Neuinstallation" ist "ok" wählen

(SuSE partitioniert automatisch den restlichen Festplattenplatz)

- j) "Partitionierung" auswählen und die Einträge `"/dev/hdc3"` und `"/dev/hdc2"` löschen
- k) erweiterte Partition und logische Laufwerke laut Aufgabenstellung einrichten und Linux installieren
- l) erkannte Graphikkarte akzeptieren (ohne 3D-Beschleunigung) und Monitordaten einstellen
- m) "Installationseinstellungen" akzeptieren und "automatische Druckererkennung" überspringen
(Linux wird anschließend im Mehrbenutzerbetrieb gestartet)
- n) melden Sie sich als "root" an, öffnen Sie ein Eingabefenster und starten Sie "yast2"
- o) konfigurieren Sie den Rechnernamen, Namensdienst usw.
- p) ändern Sie in der Datei `"/boot/grub/menu.lst"` den Eintrag "title windows" in "title msdos"
- q) erstellen Sie eine *Boot-Diskette* (**eigene Diskette mitbringen!**)
 - formatierte Diskette in das Diskettenlaufwerk einlegen
 - `/sbin/mke2fs /dev/fd0`
 - `/bin/mount /media/floppy`
 - `/bin/mkdir -p /media/floppy/boot/grub`
 - `cd /boot/grub`
 - `/bin/tar cf - ./* | (cd /media/floppy/boot/grub; tar xvf -)`
(alles in einer Zeile eingeben; stellen Sie sicher, daß Sie **keine Tippfehler** haben)
 - `/bin/umount /media/floppy`
 - `cd /`

- `/usr/sbin/grub`

geben Sie nach der Eingabeaufforderung "`grub>`" folgendes ein:

```
root (hd0,4)  
setup (fd0)  
quit
```

(Jetzt haben Sie eine *Boot*-Diskette, die Ihnen das Starten von MS-DOS und Linux erlaubt. Starten Sie beide Betriebssysteme über die *Boot*-Diskette.)

- r) die Grundinstallation von Linux ist jetzt abgeschlossen

- s) Generierung eines neuen *kernels* vorbereiten

- erstellen Sie die folgenden Dateien

- ◆ `cd /boot`
- ◆ `cp -p vmlinuz vmlinuz.old`
- ◆ `cp -p vmlinuz vmlinuz.suse`
- ◆ `cp -p System.map-<version> System.old-<version>`
- ◆ `cp -p System.map-<version> System.suse-<version>`

- ändern / erweitern Sie das Menü von *grub* auf der Festplatte und auf der *Boot*-Diskette

- ◆ Konfiguration *linux* startet `/boot/vmlinuz`
- ◆ Konfiguration *linux.old* startet `/boot/vmlinuz.old`
- ◆ Konfiguration *linux.suse* startet `/boot/vmlinuz.suse`
- ◆ Konfiguration *msdos* startet MS-DOS

(Linux kann jetzt über *linux*, *linux.old* oder *linux.suse* gestartet werden, so daß auf jeden Fall ein funktionsfähiger *Kernel* existiert.)

t) generieren Sie einen neuen *kernel*, der auf die Hardware des Rechners und die Ziele der Lehrveranstaltung abgestimmt ist (**Faustregel:** Treiber für die Hardware und die grundlegenden Systeme (*ext2fs*, *elf*- und *a.out*-Binärformat) werden in den *Kernel* eingebunden. Alle anderen Dinge werden als *Modul* generiert. Alle nicht benötigten Dinge werden nicht erzeugt.)

- `cd /usr/src/linux`
- `make clean`
- `make config` oder ***make menuconfig*** oder ***make xconfig***
- `make dep`
- `make bzlilo >&make_msg`
(*make_msg* nach *warning* und *Error* durchsuchen; *make_msg* ggf. löschen)
- neue *Boot*-Diskette erzeugen (**eigene Diskette mitbringen!**)
 - ◆ `make bzdisk`
 - ◆ `rdev -R /dev/fd0 1`
- `make modules >&make_modules_msg`
(*make_modules_msg* nach *warning* und *Error* durchsuchen; *make_modules_msg* ggf. löschen)
- `make modules_install`

u) Linux beenden / neu starten

`init 0`

`shutdown ...`

`init 6`

`reboot`

<Ctrl><Alt> bzw. <Strg><Alt><Entf>

Schaltfläche auf der graphischen Oberfläche

3. Solaris-*Boot*-Diskette einlegen, Rechner erneut starten und Solaris vom NFS-*Server swlab92* installieren (*Device [] Net*)
 - interaktive Installation wählen
 - Sprachumgebung *German-Euro*
 - Graphikkarte, Monitor, Tastatur und Maus einstellen
(MultiFrequency 95 kHz (up to 1600x1200 @ 75 Hz), 17-inch, 1024x768 - 256 colors @ 75 Hz)
 - kein IPv6, kein Kerberos, Rechner ist im Subnetzwerk, ...
 - *Entire Distribution* wählen
 - manuelles Layout der Festplatte wählen und Solaris-Partition entsprechend Aufgabenstellung einrichten
 - nach der Installation setzen Sie in der Datei */etc/rc6* ein Kommentarzeichen vor die Zeile

```
/sbin/umount /var >/dev/null 2>&1
```

(andernfalls liefern *init 0* bzw. *init 6* die folgenden Fehler:
INIT: failed write of utmpx entry: "s0" oder "s6"
INIT: failed write of utmpx entry: "fw" oder "rb")
 - Datei */etc/defaultrouter* mit Inhalt *193.174.24.65* erstellen
 - Datei */etc/nsswitch.conf* sollte u.a. die Zeile *hosts: dns files* enthalten (ggf. erstellen / ändern)
 - */etc/resolv.conf* sollte u.a. die folgenden Zeilen enthalten:

```
nameserver 193.174.25.38
nameserver 193.174.26.133
domain informatik.fh-fulda.de
```

- Netzwerkkarte konfigurieren
 - ◆ Datei `/kernel/drv/elxl.conf` im Editor öffnen
 - ◆ "#" vor den folgenden Zeilen entfernen bzw. Zeilen einfügen

```
#full-duplex=1
#speed=100
```
 - Solaris beenden / **neu starten** (init 0, shutdown ..., **init 6, reboot**)
 - Solaris aktualisieren / korrigieren (*Patch* einspielen)
 - ◆ `mkdir /mnt/remote`
 - ◆ `mount -F nfs swlab92:/export/install /mnt/remote`
 - ◆ `cd /mnt/remote/8_x86_Recommended`
 - ◆ `./install_cluster`
<y><Return-Taste>
(Fehlermeldungen ignorieren)
4. *Grub-Boot-Diskette* einlegen und System neu starten
- erweitern Sie die Datei `/boot/grub/menu.lst` um die Einträge

```
title Solaris
rootnoverify (hd0,2)
makeactive
chainloader +1
```
 - installieren Sie *grub* im MBR der Festplatte

```
grub
root (hd0,4)
setup (hd0)
quit
```
 - ändern Sie analog die Menü-Datei von *grub* auf der *Boot-Diskette* (auf der Diskette muß *grub* nicht noch einmal installiert werden)
5. Windows 2000 installieren und in *grub* einfügen (optional)

- *Kernel-Moduln in Linux*

- werden bei Bedarf geladen
- werden automatisch aus dem Betriebssystemkern entfernt, wenn sie nicht mehr benötigt werden
- befinden sich im Verzeichnis */lib/modules/<BS-Version>/...*
- Konfigurationsparameter stehen in */etc/modules.conf*
- in */lib/modules/<BS-Version>/modules.dep* werden Abhängigkeiten gespeichert
(welche andere Moduln geladen werden müssen, bevor das gewünschte Modul geladen werden darf)

```
...  
/lib/modules/2.2.13/fs/lockd.o: /lib/modules/2.2.13/misc/sunrpc.o  
/lib/modules/2.2.13/fs/nfs.o: /lib/modules/2.2.13/misc/sunrpc.o \  
/lib/modules/2.2.13/fs/lockd.o  
...  
/lib/modules/2.2.13/net/ne.o: /lib/modules/2.2.13/net/8390.o  
...  
/lib/modules/2.2.13/scsi/aic7xxx.o:  
...
```

- Kommandos
 - ◆ insmod
 - ◆ lsmod
 - ◆ rmmod
 - ◆ modprobe
 - ◆ depmod

- *Kernel*-Moduln in Solaris
 - werden bei Bedarf geladen
 - befinden sich im Verzeichnis */kernel/...* bzw. */usr/kernel/...*
 - Konfigurationsparameter stehen in *(/usr)/kernel/*/*.conf*
 - Suchpfad für Moduln kann in */etc/system* geändert werden
 - Kommandos
 - ◆ *modload*
 - ◆ *modinfo*
 - ◆ *modunload*
 - *modload* erwartet absoluten Pfad, relativen Pfad zum aktuellen Verzeichnis oder relativen Pfad zu einem Modulverzeichnis
 1. *modload /kernel/drv/nei* ⇒ Treiber *nei* aus */kernel/drv*
 2. *modload drv/nei* ⇒ Treiber *nei* aus *./drv*
 3. *modload -p drv/nei* ⇒ Treiber *nei* aus */kernel/drv*
oder */usr/kernel/drv*

1.2 Starten und Anhalten des Systems

- Rechner einschalten \Rightarrow *Boot*-Vorgang wird automatisch oder nach Eingabe eines Kommandos gestartet
- Ablauf abhängig von der Hardware (z.B. *Sparc* oder *PC*) und dem Betriebssystem-Typ (z.B. *BSD UNIX* oder *System V UNIX*)
 1. Anweisungen im NVRAM oder ROM-BIOS werden ausgeführt
 - a) Selbsttest des Systems (*Power-on self-test*)
 - b) Sparc: – *Autoboot* durchführen, falls *autoboot == true*
 - \Rightarrow Boot-Vorgang mit im NVRAM festgelegten Parametern
 - in Systemmonitor verzweigen, falls *autoboot == false* oder *Boot*-Vorgang mit Tastenkombination <Stop><a> abgebrochen wurde
 - \Rightarrow Variablen im NVRAM (Systemmonitor) können geändert werden
 - \Rightarrow spezieller *Boot*-Vorgang kann ausgewählt werden (z.B. 32- oder 64-Bit Betriebssystemkern)
 - z.B. *bootblk* aus den ersten 16 Blöcken der Festplatte lesen und starten (kann lesend auf ufs-Dateisystem zugreifen)

- PC:
- *Master-Boot-Record* einlesen und starten
 - *Boot-Block* aus aktiver Partition lesen und starten

2. Betriebssystemkern laden

- Sparc:
- das *bootblk*-Programm lädt ein weiteres Programm, das von der Architektur des Rechners abhängt (z.B. bei einer Ultra-Sparc: `/platform/sun4u/ufsboot`)
 - *ufsboot* lädt und startet den Betriebssystemkern, der aus einem Hardware-abhängigen und einem Hardware-unabhängigen Teil besteht
 - ◆ abhängiger Teil:
z.B. `/platform/sun4u/kernel/unix` (32 Bit)
oder `/platform/sun4u/kernel/sparcv9/unix` (64 Bit)
 - ◆ unabhängiger Teil:
`/platform/sun4u/kernel/genunix` (generic unix) bzw.
`/platform/sun4u/kernel/sparcv9/genunix`
 - der *unix-kernel* übernimmt die weitere Initialisierung
 - ◆ unbedingt benötigte *Kernel*-Moduln laden
 - ◆ Systemprozesse starten

PC: – Solaris 7 x86

- ◆ das Partitions-*Boot-Block*-Programm führt einige Initialisierungen durch und liest dann das *bootblk*-Programm aus den ersten Blöcken der Solaris-Partition
- ◆ falls sich auf der Festplatte auch andere Betriebssysteme befinden, startet *bootblk* den *bootmanager* (falls nur Solaris auf der Festplatte ist, wird er nicht gestartet)
- ◆ falls Solaris gestartet werden soll, wird `/platform/i86pc/boot/solaris/boot.bin` gestartet
- ◆ *boot.bin* initialisiert die Hardware und schaltet die CPU in den *Protected-Mode*
- ◆ danach führt es das Skript `/platform/i86pc/boot/solaris/boot.rc` aus, das den Konfigurationsassistenten *bootconf.exe* im selben Verzeichnis startet
 - ⇒ der Wert von *auto_boot* entscheidet, ob der *Boot*-Vorgang automatisch fortgesetzt wird
 - ⇒ mit <Esc> kann der *Boot*-Vorgang unterbrochen und in den Konfigurationsassistenten gewechselt werden
 - ⇒ ggf. Hardware-Anpassungen durchführen

- ◆ nachdem der Konfigurationsassistent geendet hat, erhält *boot.bin* die Kontrolle zurück
- ◆ *boot.bin* ruft das Skript */etc/bootrc* auf, über das der Solaris-Boot-Interpreter gestartet werden kann (scheint Bestandteil von *boot.bin* zu sein)

⇒ jetzt ist man in etwa an der Stelle 1. b)
(der Boot-Interpreter ist eine Art Ersatz für das NVRAM bei Sparc-Rechnern)
- ◆ anschließend wird *ufsboot* ausgeführt, das den *Kernel* lädt (vermutlich ebenfalls Bestandteil von *boot.bin*)
(*/platform/i86pc/kernel/unix*, */kernel/genunix*)
- ◆ der *unix-kernel* führt dann wieder die weitere Initialisierung durch, lädt die *Kernel*-Moduln und startet einige Systemprozesse

– Linux

- ◆ ähnlich aufwendig wie bei Solaris x86
(siehe: Rubini, A.: *Linux Device Drivers*. O'Reilly, 1998, S. 362ff)
- ◆ lilo benutzt BIOS-Routinen und muß deshalb die Blöcke auf der Festplatte kennen, die das Betriebssystem enthalten
(es wird eine Blocktabelle angelegt, die nach jeder Änderung des Betriebssystemkerns aktualisiert werden muß)

3. einige Systemprozesse starten

(Reihenfolge und Namen abhängig von der Betriebssystemversion)

Solaris (Prozesse 0-3):

```

UID    PID    PPID    C     STIME TTY          TIME CMD
root    0      0      0  12:34:48 ?           0:00 sched
root    1      0      0  12:34:48 ?           0:00 /etc/init -
root    2      0      0  12:34:48 ?           0:00 pageout
root    3      0      0  12:34:48 ?           0:00 fsflush
...
root   159     1      0  12:35:23 ?           0:00 /usr/sbin/inetd -s
...
root   254     1      0  12:35:39 console  0:00 /usr/lib/saf/ttymon -g ...

```

Linux (Prozesse 1-5):

```

PID TTY          STAT     TIME COMMAND
  1 ?            S        0:04 init HOME=/ TERM=linux BOOT_IMAGE=linux
  2 ?            SW       0:00 [kflushd]
  3 ?            SW       0:00 [kupdate]
  4 ?            SW       0:00 [kpiod]
  5 ?            SW       0:00 [kswapd]
...
143 ?           SW       0:00 [inetd]
...
210 tty1        SW       0:00 [login]
211 tty2        SW       0:00 [mingetty]
...

```

4. *init*-Prozeß initialisiert Prozeßsystem (letzter Schritt beim Boot-Vorgang)

- abhängig vom Betriebssystem-Typ (BSD, System V)
- hier wird nur die Variante für *UNIX System V* betrachtet
- Initialisierung wird über Datei */etc/inittab* gesteuert
(Tabelleneinträge sind abhängig von der Betriebssystemversion)

- */etc/inittab* bei Solaris x86

```
ap::sysinit:/sbin/autopush -f /etc/iu.ap
ap::sysinit:/sbin/soconfig -f /etc/sock2path
fs::sysinit:/sbin/rcS sysinit >/dev/console 2<>/dev/console </dev/console
is:3:initdefault:
p3:s1234:powerfail:/usr/sbin/shutdown -y -i5 -g0 >/dev/console
                                     2<>/dev/console
sS:s:wait:/sbin/rcS                   >/dev/console 2<>/dev/console </dev/console
s0:0:wait:/sbin/rc0                   >/dev/console 2<>/dev/console </dev/console
s1:1:respawn:/sbin/rc1                 >/dev/console 2<>/dev/console </dev/console
s2:23:wait:/sbin/rc2                  >/dev/console 2<>/dev/console </dev/console
s3:3:wait:/sbin/rc3                   >/dev/console 2<>/dev/console </dev/console
s5:5:wait:/sbin/rc5                   >/dev/console 2<>/dev/console </dev/console
s6:6:wait:/sbin/rc6                   >/dev/console 2<>/dev/console </dev/console
fw:0:wait:/sbin/uadmin 2 0             >/dev/console 2<>/dev/console </dev/console
of:5:wait:/sbin/uadmin 2 6             >/dev/console 2<>/dev/console </dev/console
rb:6:wait:/sbin/uadmin 2 1             >/dev/console 2<>/dev/console </dev/console
sc:234:respawn:/usr/lib/saf/sac -t 300
co:234:respawn:/usr/lib/saf/ttymon -g -h -p "`uname -n` console login: "
                                     -T AT386 -d /dev/console -l console
```

(alle Angaben erfolgen in einer Zeile. Umbruch bei "p3:" und "co:" nur wegen Lesbarkeit)

- */etc/inittab* bei Linux

```
# default runlevel
id:2:initdefault:

# check system on startup
# first script to be executed if not booting in emergency (-b) mode
si:I:bootwait:/sbin/init.d/boot

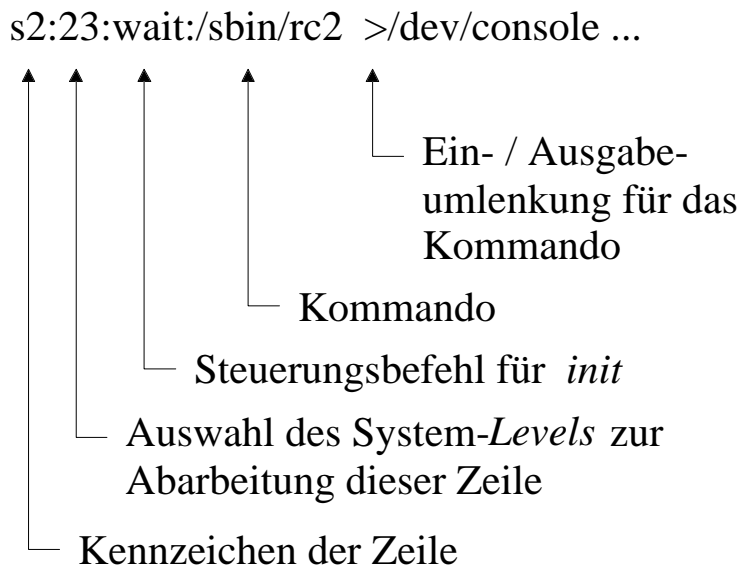
10:0:wait:/sbin/init.d/rc 0
11:1:wait:/sbin/init.d/rc 1
12:2:wait:/sbin/init.d/rc 2
13:3:wait:/sbin/init.d/rc 3
#14:4:wait:/sbin/init.d/rc 4
#15:5:wait:/sbin/init.d/rc 5
16:6:wait:/sbin/init.d/rc 6

# what to do in single-user mode
ls:S:wait:/sbin/init.d/rc S
~~:S:respawn:/sbin/sulogin

# what to do when CTRL-ALT-DEL is pressed
ca::ctrlaltdel:/sbin/shutdown -r -t 4 now

...
# getty-programs for the normal runlevels
# <id>:<runlevels>:<action>:<process>
# The "id" field MUST be the same as the last
# characters of the device (after "tty").
1:123:respawn:/sbin/mingetty --noclear tty1
2:123:respawn:/sbin/mingetty tty2
...
```

- Aufbau der Tabelle



- Systemzustände

Zustand	Solaris	Linux (SuSE 7.3)
0	<i>Shutdown</i>	<i>Shutdown</i>
1	Einbenutzersystem	Einbenutzersystem
2	Mehrbenutzersystem	Mehrbenutzersystem ohne Netzwerk
3	Mehrbenutzersystem mit Netzwerkbetrieb	Mehrbenutzersystem mit Netzwerk
4	frei	frei
5	<i>Powerdown</i>	Mehrbenutzersystem mit Netzwerk und XDM
6	<i>Reboot</i>	<i>Reboot</i>
s, S	Einbenutzersystem	Einbenutzersystem

- über den *init*-Befehl kann der Systemzustand geändert werden
- Pseudozustände *a*, *b* und *c* bei Solaris und Linux
 - ◆ der Systemzustand wird nicht geändert
 - ◆ die Zeilen für den Pseudozustand in der Datei */etc/inittab* werden ausgeführt
 - ◆ hiermit können z.B. zusätzliche Hintergrundprozesse gestartet werden
- mit *init q* oder *init Q* kann der *init*-Prozeß veranlaßt werden, die Datei */etc/inittab* erneut auszuwerten
(Änderungen in der Datei können auf diese Weise ohne Systemneustart gültig werden)
- enthält die zweite Spalte in der Datei */etc/inittab* keinen Systemzustand, gilt die Zeile für alle Zustände

- Steuerungsbeefehle zur Abarbeitung der Zeilen in */etc/inittab*

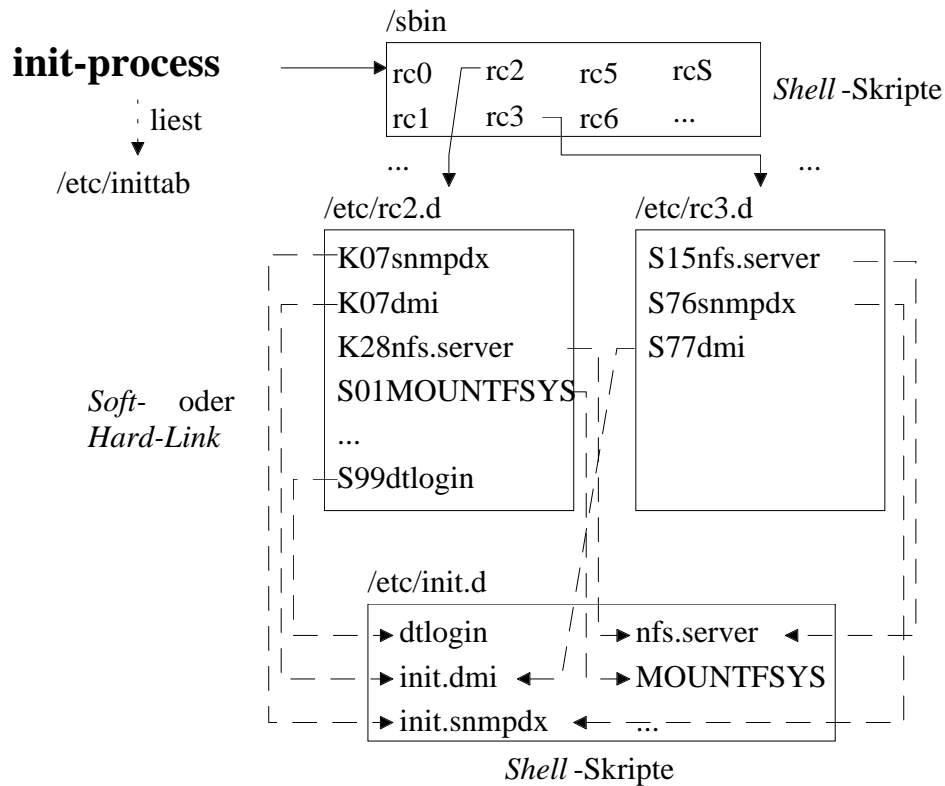
Steuerungs- befehl	Bedeutung
initdefault	gibt an, welcher Zustand beim <i>Boot</i> -Vorgang eingenommen wird; falls das Systemzustandsfeld leer ist (entspricht bei Solaris 0123456) oder den Wert 6 enthält, wird der <i>Boot</i> -Vorgang bei Solaris in einer Endlosschleife wiederholt; falls diese Zeile fehlt, fragt <i>init</i> nach dem einzunehmenden Systemzustand
boot	Zeile wird nur beim <i>Boot</i> -Vorgang ausgeführt; bei Linux wird das Systemzustandsfeld ignoriert; auf das Ende des Prozesses wird nicht gewartet; diese Zeilen werden vor allen anderen Zeilen der <i>/etc/inittab</i> abgearbeitet
bootwait	Zeile wird ausgeführt, wenn das System beim <i>Boot</i> -Vorgang vom Zustand <i>single-user</i> in den Zustand <i>multi-user</i> wechselt; bei Linux wird das Systemzustandsfeld ignoriert; auf das Ende des Prozesses wird gewartet
sysinit	Zeile wird ausgeführt, bevor <i>init</i> auf die Konsole zugreift; dieser Steuerungsbeefehl sollte nur benutzt werden, um die Konsole zu initialisieren, damit <i>init</i> ggf. nach dem einzunehmenden <i>run-level</i> fragen kann; warten bis Programm beendet ist, bevor weitere Zeilen bearbeitet werden; unter Linux werden diese Zeilen vor allen <i>boot</i> - und <i>bootwait</i> -Zeilen ausgeführt
wait	auf Programmende warten; Zeile wird nur beim Betreten des <i>run-levels</i> ausgeführt und sonst ignoriert
respawn	Prozeß wird nach Beendigung wieder gestartet
ctrlaltdel	Zeile wird ausgeführt, wenn <Ctrl><Alt> bzw. <Strg><Alt><Entf> gedrückt wird, so daß <i>init</i> das Signal SIGINT erhält; nur bei Linux
powerfail	Zeile wird ausgeführt, wenn <i>init</i> das Signal SIGPWR erhält
...	siehe z.B. Ausgabe von <i>man inittab</i>

- *init* startet die in */etc/inittab* festgelegten Programme / *Shell*-Skripte
 - Solaris: spezielle Skripte für jeden Systemzustand
 - Linux: i.w. zwei Skripte, die über Parameter gesteuert werden

- einige Aufgaben der *Shell*-Skripte
 - Umgebungsvariablen initialisieren
 - ◆ Suchpfad für Programme einstellen
 - ◆ Namen des Rechners setzen
 - ◆ Zeitzone setzen
 - ◆ ...
 - Dateisysteme überprüfen (*fschk*)
 - Dateisysteme einbinden (*mount*)
 - ggf. temporäre Dateien löschen (*/tmp*)
 - Netzwerk konfigurieren
 - Dämon-Prozesse und Netzwerkdienste starten
 - ggf. Abrechnungs- und Überwachungsprogramme aktivieren (*accounting, disk quota*)
 - ...

- Steuerung des Systemzustands unter Solaris
 - für jeden Zustand (*run-level*) *n* gibt es ein spezielles Skript *rcn*
 - für jedes Skript *rcn* gibt es ein Verzeichnis *rcn.d*, in dem sich die Skripte *KNumName* und *SNumName* befinden
 - Skripte *KNumName* beenden Prozesse, die im neuen Zustand nicht mehr benötigt werden und Skripte *SNumName* starten Prozesse für den Zustand
 - *Num* gibt die Reihenfolge an, in der die Skripte ausgeführt werden
 - in den Verzeichnissen *rcn.d* stehen nur Verweise (*links*) auf die Skripte, so daß der Administrator einen Prozeß leicht aus einem Zustand entfernen kann, ohne das Skript zu löschen
(besser: den Verweis K... bzw S... nicht löschen, sondern in xK... bzw. xS... umbenennen ⇒ da die Skripte *rcn* nur nach Dateien suchen, die mit K oder S beginnen, werden die Prozesse auf diese Weise auch nicht gestoppt bzw. gestartet)
 - die eigentlichen Skripte stehen im Verzeichnis *init.d*
 - der Administrator kann eigene Skripte in *init.d* hinzufügen und dann die entsprechenden Verweise in *rcn.d* einfügen
(eigene Skripte sollten i.a. erst nach allen Systemskripten ausgeführt werden, damit ein korrektes Hochfahren des Systems gewährleistet bleibt)
 - der Administrator kann die Skripte in *init.d* an lokale Gegebenheiten anpassen
(**Äußerste Vorsicht!** U.U. startet das System danach nicht mehr. U.U. funktionieren einige Programme nicht mehr, weil die Initialisierung des Systems unvollständig ist.)

- Zusammenspiel der *Shell*-Skripten beim Starten von Solaris



- der Verweis */etc/rc2.d/S99dtlogin* sorgt z.B. dafür, daß automatisch die graphische Oberfläche CDE gestartet wird

(Falls *S99dtlogin* in *xS99dtlogin* umbenannt wird, würde Solaris in eine kommandozeilenorientierte Oberfläche starten. Nach einem *LOGIN* würden dann z.B. wie früher unter *Open Windows* die Dateien *~/.cshrc* und *~/.login* ausgeführt werden.)

- **/sbin/rc2**

```
#!/sbin/sh
...
#ident  "@(#)rc2.sh      1.15      98/05/10  SMI"

# Run Commands executed when the system is changing to init state 2,
# traditionally called "multi-user". Pick up start-up packages for mounts,
# daemons, services, etc.
PATH=/usr/sbin:/usr/bin

# Export boot parameters to rc scripts
set -- ` /usr/bin/who -r `

_INIT_RUN_LEVEL="$7"      # Current run-level
```

```

_INIT_RUN_NPREV="$8"    # Number of times previously at current run-level
_INIT_PREV_LEVEL="$9"  # Previous run-level

set -- ` /usr/bin/uname -a `

_INIT_UTS_SYSNAME="$1" # Operating system name (uname -s)
_INIT_UTS_NODENAME="$2" # Node name (uname -n)
...
if [ $_INIT_PREV_LEVEL = S -o $_INIT_PREV_LEVEL = 1 ]; then
    echo 'The system is coming up. Please wait.'
elif [ $_INIT_RUN_LEVEL = 2 ]; then
    echo 'Changing to state 2.'
    if [ -d /etc/rc2.d ]; then
        for f in /etc/rc2.d/K*; do
            if [ -s $f ]; then
                case $f in
                    *.sh) . $f ;;
                    *)   /sbin/sh $f stop ;;
                esac
            fi
        done
    fi
fi
if [ $_INIT_PREV_LEVEL != 2 -a $_INIT_PREV_LEVEL != 3 -a -d /etc/rc2.d ];
then
    for f in /etc/rc2.d/S*; do
        if [ -s $f ]; then
            case $f in
                *.sh) . $f ;;
                *)   /sbin/sh $f start ;;
            esac
        fi
    done
fi
...

```

- /etc/rc2.d/S74autofs

```

#!/sbin/sh
#
# Copyright (c) 1993, 1997 by Sun Microsystems, Inc.
# All rights reserved.
#
#ident "@(#)autofs 1.5 97/12/08 SMI"

case "$1" in
'start')
    /usr/lib/autofs/automountd </dev/null >/dev/console 2>&1
    /usr/sbin/automount &
    ;;

'stop')
    /sbin/umountall -F autofs
    /usr/bin/pkill -x -u 0 automountd
    ;;

*)
    echo "Usage: $0 { start | stop }"
    ;;
esac
exit 0

```

- Steuerung des Systemzustands unter Linux (SuSE 7.3)
 - YaST schreibt alle Angaben zur Konfiguration in `/etc/rc.config`, `/etc/rc.config.d/*` und `/etc/modules.conf`

- ◆ */etc/rc.config*

```
KEYTABLE="de-lat1-nd.map.gz"
LANGUAGE="german"

# Some people don't want SuSEconfig to modify the system. With
# this entry you can disable SuSEconfig completely.
# Please don't contact our support if you have trouble configuring
# your system after having disabled SuSEconfig. (yes/no)
#
ENABLE_SUSECONFIG=yes
...
# networking
# number of network cards: "_0" for one, "_0 _1 _2 _3" for four
# cards
#
NETCONFIG="_0"

# IP Adresses
#
IPADDR_0="193.174.24.121"
IPADDR_1=""
IPADDR_2=""
IPADDR_3=""

# network device names (e.g. "eth0")
#
NETDEV_0="eth0"
...
```

- ◆ */etc/modules.conf*

```
# Aliases - specify your hardware
alias eth0 ne
alias scsi_hostadapter off
...
options ne          io=0x2a0 irq=10
...
```

- ◆ danach wird automatisch `/sbin/SuSEconfig` aufgerufen, das die Änderungen ggf. in weitere Konfigurationsdateien einträgt

(Die Dateien können auch von Hand im Zustand *single-user* mit einem Editor geändert werden. Danach muß unbedingt `/sbin/SuSEconfig` aufgerufen werden.)

- alle Initialisierungsskripte lesen zuerst die Datei */etc/rc.config*, um die aktuellen Werte zu erfahren
- zuerst wird das Skript */etc/init.d/boot* aufgerufen

```
#!/bin/sh
...
# first script to be executed from init on system startup

. /etc/rc.config
echo "Running $0"

ECHO_RETURN=$rc_done
echo -n "Mounting /proc device"
mount -n -t proc proc /proc || ECHO_RETURN=$rc_failed
echo -e "$ECHO_RETURN"

# possibly there are file systems on devices, which need a kernel
# module to be loaded.  So start kerneld as early as possible.
#
if test "$START_KERNELD" = yes -a \
    -x /sbin/kerneld -a ! -e /proc/sys/kernel/modprobe ; then
    echo -n "Starting kerneld:"
    ECHO_RETURN=$rc_done_up
    /sbin/kerneld || ECHO_RETURN=$rc_failed_up
    echo -e "$ECHO_RETURN"
fi
...
```

- danach wird das Skript */etc/init.d/rc* aufgerufen, dem der gewünschte Systemzustand als Parameter übergeben wird

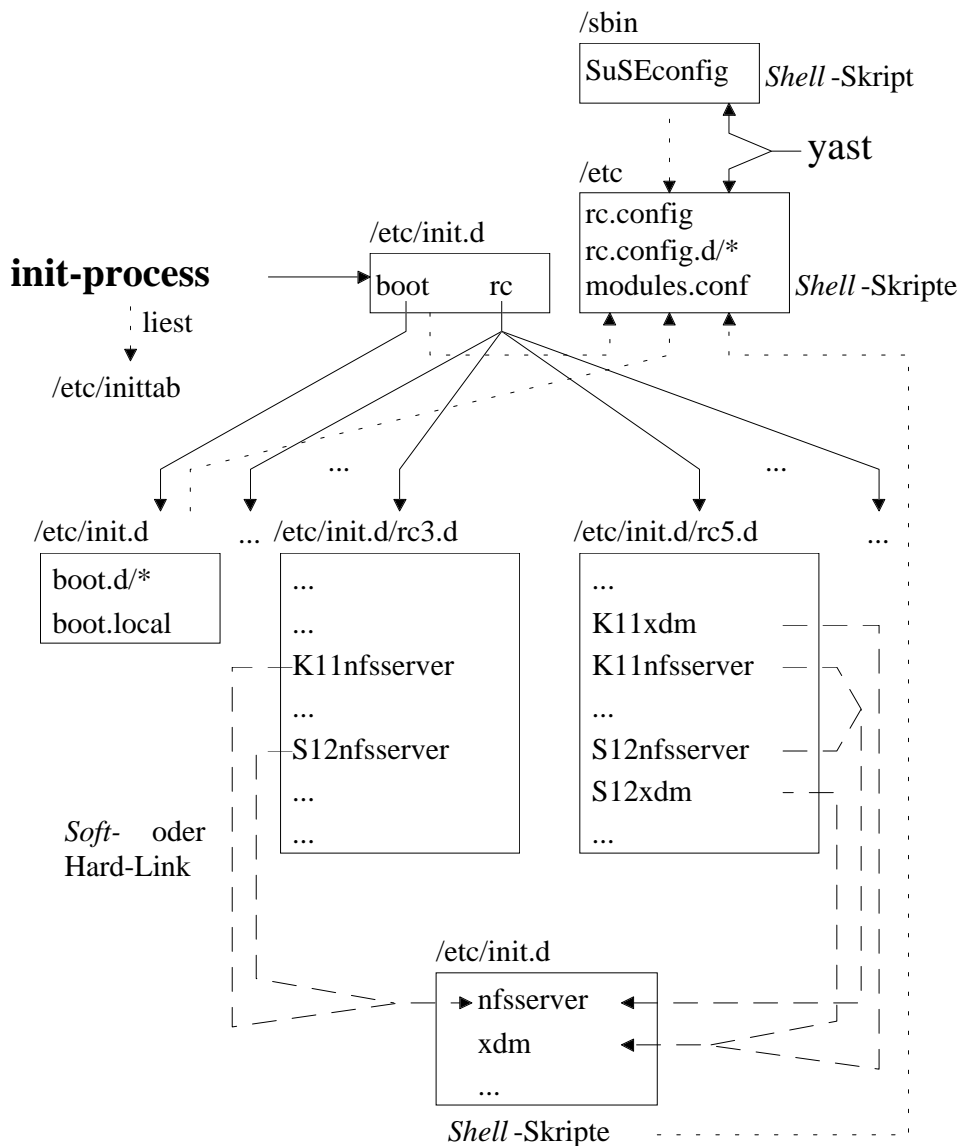
```
#!/bin/bash
...
# This file is responsible for starting/stopping services
# when the runlevel changes.  If the action for a particular
# feature in the new run-level is the same as the action in
# the previous run-level, this script will neither start nor
# stop that feature.
...
. /etc/rc.config
...
#
# run the KILL scripts of the previous runlevel
#
for i in $prevdir/K*; do
    test -x "$i" || continue
    # don't run the kill script, if the new runlevel has start script
    start=${i##*/K$rex}
    set -- $curdir/S$rex$start
    test -n "$2" && echo -e "$0: more than one start link\n -- @"
    test -f $1 && continue
    $i stop || failed="{failed} ${start}"
    echo -n -e "$rc_reset"
done
```

```
# run the START scripts of the current (new) runlevel
#
for i in $curdir/s*; do
    test -x "$i" || continue

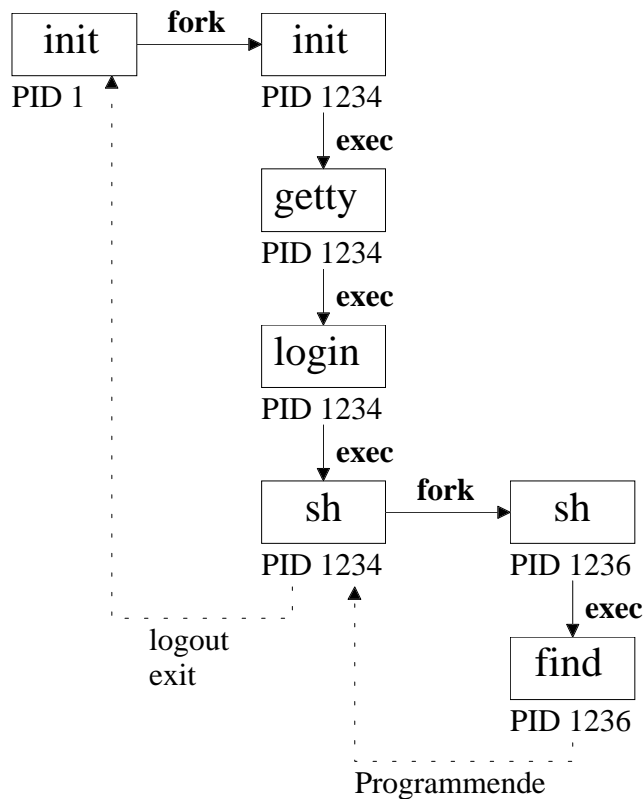
    # don't run start script, if previous runlevel had a kill script
    start=${i##*/S$rex}
    set -- $prevdir/K$rex$start
    test -n "$2" && echo -e "$0: more than one kill link\n -- $@"
    test -f $1 && continue

    $i start || failed="${failed} ${start}"
    echo -n -e "$rc_reset"
done
...
```

- **Zusammenspiel der Shell-Skripten beim Starten von Linux**
(SuSE-Distribution 7.3, siehe auch Datei "/etc/init.d/README")



- Generierung von LOGIN-Prozessen (bei Solaris etwas komplizierter)



- Anhalten des Systems

- shutdown [Optionen] [<Meldungstext>]
- halt [Optionen]
- poweroff [Optionen]
- init 0
- *init 6* oder *reboot* [Optionen] (stoppen und sofort wieder starten)
- vergewissern Sie sich, daß Sie den richtigen Rechner anhalten
- vergewissern Sie sich, daß sich alle Benutzer abgemeldet haben
- alle *Server*-Dienste des Rechners stehen anschließend nicht mehr zur Verfügung

1.3 UNIX-Systembefehle

- im folgenden wird eine kleine Auswahl von Kommandos aus der Vielfalt der UNIX-Kommandos in alphabetischer Reihenfolge vorgestellt
- ergänzen Sie die Auflistung durch einige wichtige Optionen der Kommandos unter Linux und Solaris
- berücksichtigen Sie insbesondere, ob sich die Kommandos in den Optionen unterscheiden
- berücksichtigen Sie auch, ob eine bestimmte Aktion durch unterschiedliche Kommandos oder Optionen erreicht wird
- benutzen Sie "n.v.", falls ein Kommando nicht verfügbar ist
- tragen Sie im Feld "Optionen" ein anderes Kommando ein, wenn die Aktion durch ein anderes Kommando zur Verfügung gestellt wird

Befehl	einige Optionen		Kurzbeschreibung
	Linux	Solaris	
apropos			Kommandosuche nach Schlüsselwort
ar			Bibliothek erstellen / bearbeiten
as			Assembler
at			Kommando zu vorgegebener Zeit starten
atq			<i>at</i> -Auftragsliste anzeigen
atrm			Aufträge aus <i>at</i> -Auftragsliste löschen
awk			Reportgenerator
basename			Dateinamen extrahieren
bg			Vordergrundprozeß in Hintergrund legen
cancel			Druckaufträge löschen
cat			Dateiinhalte ausgeben
catman			Indexdatei für Handbuchseiten erstellen
cc			C Compiler
CC			C++ Compiler
cd			neues Arbeitsverzeichnis wählen
chgrp			Gruppenzugehörigkeit ändern
chmod			Zugriffsrechte ändern
chown			Eigentümer ändern
col			Filter (z.B. "man xyz col -b >xyz.txt")
compress			Datei komprimieren

Befehl	einige Optionen		Kurzbeschreibung
	Linux	Solaris	
constype			gibt den Typ der Graphikkarte aus
cp			Datei kopieren
cpio			Dateien sichern / wiederherstellen
cron			automatische Wiederholung von Prozessen
crontab			Auftragsdatei für <i>cron</i>
csch			C-Shell
csplit			Datei in mehrere Dateien aufteilen
date			Datum / Uhrzeit ausgeben / setzen
dd			kopieren / konvertieren von Dateien
devinfo			Daten über Aufbau der Festplatte anzeigen
df			Anzahl freier Blöcke auf Datenträger
dfmounts			Info zu eingebundenen Dateisystemen
dfshares			Info zu freigegebenen Dateisystemen
diff			Unterschied zwischen zwei Dateien
dircmp			Unterschiede zwischen zwei Verzeichnissen
dirname			Pfadnamen extrahieren
disable			deaktiviert einen Drucker
dmesg			gibt Systemmeldungen aus
drvconfig			Neukonfiguration der Geräte
du			Plattenbelegung durch Dateien/Verzeichnisse

Befehl	einige Optionen		Kurzbeschreibung
	Linux	Solaris	
dump			Teile einer Objektdatei ausgeben
echo			Ausgabe auf Bildschirm
egrep			Suche in Dateien
eject			Datenträger auswerfen (Diskette, CDROM)
enable			aktiviert einen Drucker
env			gibt Umgebungsvariablen aus
exit			<i>Shell</i> beenden
expand			Tabulator- in Leerzeichen wandeln
export			<i>Shell</i> -Variable als Umgebungsvariable exportieren (global \Rightarrow kann vererbt werden)
fdformat			Diskette formatieren
fdisk			Aufteilung der Festplatte (nur PC)
ffbconfig			Konfiguration einer <i>Creator</i> -Graphikkarte
fg			Hintergrundprozeß in Vordergrund holen
fgrep			Suche in Dateien
file			Klassifizierung von Dateien
find			sucht Dateien mit vorgegebenen Attributen
finger			Informationen zu Benutzern
format			Festplatte konfigurieren
fsck			Konsistenzprüfung des Dateisystems

Befehl	einige Optionen		Kurzbeschreibung
	Linux	Solaris	
fstyp			Dateisystemtyp ausgeben (u.U. vor <i>mount</i>)
ftp			Dateiübertragung zwischen Rechnern
fuser			wer benutzt die Datei / das Verzeichnis?
grep			Suche in Dateien
groupadd			neue Gruppe im System eintragen
groupdel			Gruppendefinition aus System entfernen
groupmod			Gruppendefinition ändern
groups			zu welchen Gruppen gehört ein Benutzer?
grpck			Konsistenz von <i>/etc/group</i> überprüfen
halt			Prozessor anhalten
hash			<i>Hash</i> -Tabelle der <i>Bourne-Shell</i> aktualisieren
head			liefert die ersten Zeilen einer Datei
hostid			liefert / setzt Rechneridentifikation
hostname			liefert / setzt den Rechnernamen
id			Benutzer- / Gruppennummer ausgeben
ifconfig			konfiguriert den Netzanschluß
infocmp			Terminalbeschreibung ausgeben
init			Systemzustand ändern
iostat			erstellt Statistik über E/A-Aktivität
ipcrm			entfernt IPC-Einträge

Befehl	einige Optionen		Kurzbeschreibung
	Linux	Solaris	
ipcs			zeigt IPC-Einträge an
kdmconfig			<i>keyboard-display-mouse</i> konfigurieren
kill			bricht i.a. einen Prozeß ab
last			gibt An-/Abmeldungen von Benutzern aus
ld			Binder
ldd			dynamische Abhängigkeiten anzeigen
listusers			gibt alle <i>Login</i> -Namen aus
ln			<i>Hard-</i> oder <i>Soft-Link</i> erstellen
logins			Informationen über Benutzer
logname			gibt <i>login</i> -Namen des Benutzers aus
logout			<i>Login-Shell</i> beenden
lp			Datei drucken
lpadmin			Verwaltungsprogramm für <i>lp-print-spooler</i>
lpstat			Status der Druckaufträge anzeigen
ls			Dateien eines Verzeichnisses anzeigen
m4			Makro-Prozessor
mail			empfangen / absenden von <i>E-Mail</i>
mailtool			graphische Oberfläche zu <i>mail</i>
make			automatische Programmerstellung
man			Kommandobeschreibung ausgeben

Befehl	einige Optionen		Kurzbeschreibung
	Linux	Solaris	
mkdir			Verzeichnis erstellen
mkfs			Dateisystem anlegen
mknod			spezielle Datei erzeugen
modinfo			Informationen zu Moduln
modload			Modul laden
modunload			Modul entfernen
more			seitenweise Ausgabe auf Bildschirm
mount			Dateisystem in Verzeichnis einbinden
mountall			alle angegebenen Dateisysteme einbinden
mpstat			Statistik über Verteilung der Prozesse auf Prozessoren ausgeben
mt			Steuerung von Magnetbandgeräten
mv			Datei / Verzeichnis umbenennen
mvdir			Verzeichnis umbenennen
ndd			Konfigurationsparameter der Treiber setzen / anzeigen
netstat			Status eines Netzwerks anzeigen
newfs			<i>ufs</i> -Dateisystem anlegen
newgrp			Gruppennummer ändern
news			gibt Nachrichten des Administrators aus
nfsstat			NFS-Zugriffsstatistik

Befehl	einige Optionen		Kurzbeschreibung
	Linux	Solaris	
nice			Priorität eines Prozesses ändern
niscat			gibt Dateien von NIS+ aus
nisspasswd			Paßwortänderung in NIS+
nm			Symboltabelle einer Objektdatei ausgeben
passwd			Paßwortänderung
pbind			Prozeß an Prozessor binden
ping			Netzverbindung testen
pkgadd			Software-Paket hinzufügen
pkginfo			Informationen zu Software-Paket
pkgrm			Software-Paket entfernen
pldd			dynamische Bibliotheken, die Prozeß benutzt
pmap			zeigt Speicherbelegung eines Prozesses an
prtconf			anzeigen der Konfiguration
prtdiag			Hardware-Status anzeigen
prvtoc			Informationen zur Festplattenbelegung
ps			Liste aktiver Prozesse
psig			zeigt Reaktion des Prozesses auf Signale
psradm			Prozessor ab- / anschalten
psrinfo			Informationen über Prozessoren
psrset			Prozessorgruppen definieren

Befehl	einige Optionen		Kurzbeschreibung
	Linux	Solaris	
pstack			zeigt Stackbelegung des Prozesses an
ptree			zeigt Prozeßgenerationsfolge an
pwck			Konsistenz von <i>/etc/passwd</i> überprüfen
pwd			Name des aktuellen Verzeichnisses
quot			erstellt Plattenbelegungsliste
quota			vom Benutzer belegter Plattenplatz
rcp			<i>remote copy</i>
rdist			Dateiverteilung / -aktualisierung im Netz
rehash			<i>Hash</i> -Tabelle der <i>C-Shell</i> aktualisieren
rexec			<i>remote execution</i>
rewind			Magnetband zurückspulen
rlogin			<i>remote login</i>
rm			Dateien / Verzeichnisse löschen
rmdir			leeres Verzeichnis löschen
route			<i>Routing</i> -Tabelle manuell verändern
rpcinfo			anzeigen / ändern der Dienste an <i>RPC-Ports</i>
rsh			<i>remote shell</i>
runacct			Auswertung der <i>Accounting</i> -Dateien
rusers			<i>remote users</i>
rwho			<i>remote who</i>

Befehl	einige Optionen		Kurzbeschreibung
	Linux	Solaris	
sac			Zugangskontrollsystem
sag			Diagramm der Systemaktivitäten
sar			Bericht über Systemaktivitäten
sdb			Symbolischer <i>Debugger</i>
sdiff			Dateiunterschiede nebeneinander ausgeben
sed			<i>Stream</i> -Editor
sendmail			zentrales Postamt
set			<i>Shell</i> -Variablen setzen / anzeigen
setenv			Umgebungsvariablen setzen / anzeigen
sh			Bourne-Shell
share			lokale Verzeichnisse freigeben
shareall			alle festgelegten Dateisysteme freigeben
showmount			eingebundene Dateisysteme anzeigen
showrev			Informationen über Hard- und Software
shutdown			System herunterfahren
size			Größe des Text-, Daten- und <i>Stack</i> -Segments
sleep			Programm für eine Zeit suspendieren
snoop			Überwachung des Netzverkehrs
sort			Dateiinhalte sortieren
split			Datei in mehrere Dateien aufteilen

Befehl	einige Optionen		Kurzbeschreibung
	Linux	Solaris	
spray			IP-Pakete in das Netz schicken (Fehlersuche)
strace			Systemaufrufe / Signale verfolgen (Linux)
strings			ASCII-Zeichen aus Binärdatei ausgeben
strip			Symboltabelle aus Objektdatei entfernen
stty			abfragen / setzen von <i>Terminal</i> -Parametern
su			auf neue Benutzerkennung umschalten
sync			Blockpuffer auf Festplatte schreiben
sysadm			zeichenorientierte Systemverwaltung
sysadmin			wie <i>sysadm</i>
sysdef			ermittelt Systemkonfiguration
sys-unconfig			Konfigurationsinformationen auf Standardwerte zurücksetzen
tail			liefert die letzten Zeilen einer Datei
talk			erlaubt Dialog zwischen zwei Benutzern
tar			Dateien sichern / wiederherstellen
tcsh			T-C-Shell
tee			dupliziert den Text der Standardeingabe
telnet			baut Verbindung zu anderem Rechner auf
test			vergleicht Dateieigenschaften/Zeichenfolgen
tic			erzeugt <i>Terminfo</i> -Eintrag

Befehl	einige Optionen		Kurzbeschreibung
	Linux	Solaris	
time			mißt die Ausführungszeit eines Programms
touch			ändert das Änderungsdatum einer Datei
tr			Konvertierung von Zeichen
traceroute			Laufweg von Internet-Paketen
truss			Verfolgung von Systemaufrufen
ufsdump			Sicherung einer Partition
ufsrestore			<i>ufs</i> -Partition wiederherstellen
umask			Zugriffsrechte für neue Dateien
umount			Dateisystem aus Verzeichnis aushängen
umountall			alle Dateisysteme aushängen (ohne /)
uname			Systeminformationen ausgeben
uncompress			Datei dekomprimieren
unexpand			Leerzeichen in Tabulatorzeichen wandeln
uniq			entfernt Mehrfachzeilen in Dateien
unset			<i>Shell</i> -Variable löschen
unsetenv			Umgebungsvariablen löschen
unshare			hebt Freigabe von lokalen Ressourcen auf
unshareall			hebt Freigabe von lokalen Ressourcen auf
user			gibt Liste der aktiven Benutzer aus
useradd			neuen Benutzer einrichten

Befehl	einige Optionen		Kurzbeschreibung
	Linux	Solaris	
userdel			Benutzer aus System entfernen
usermod			Verwaltungsinfo eines Benutzers ändern
vi			Editor
vmstat			Statistik über Speicher- und Prozessor- auslastung
volcheck			Diskette / CDROM in Dateisystem einbinden
wall			schickt Nachricht an alle anderen Benutzer
wc			zählt Zeichen, Wörter, Zeilen einer Datei
whereis			gibt u.a. alle Pfade eines Programms aus
which			gibt den ersten Fundort des Programms aus
who			Liste der angemeldeten Benutzer ausgeben
whoami			aktuelle effektive Benutzerkennung
whodo			gibt Liste der Aktivitäten der Benutzer aus
whois			Angaben zu einem Benutzer
write			schickt Nachricht an anderen Benutzer
wsinfo			Informationen über den Rechner
xargs			baut Argumentenliste auf und führt Kom- mando mit dieser Liste aus
xhost			steuert Zugriffskontrolle auf <i>X-Server</i>
xset			anzeigen / setzen der <i>X-Server</i> -Optionen
xsetroot			Hintergrundbild bei X11-Bildschirm setzen

Befehl	einige Optionen		Kurzbeschreibung
	Linux	Solaris	
xterm			emuliert zeichenorientierten Bildschirm
xwd			erzeugt Bildschirmabzug unter X11
ypcat			gibt Dateien von NIS aus
yppasswd			Paßwortänderung in NIS

Aufgabe 1-2:

Lösen Sie die folgenden Aufgaben jeweils unter Solaris und Linux.

- Welche dynamischen Bibliotheken benutzen die Programme *ls*, *more*, *xwd* und *ypcat*?
- Welche Funktionen sind in den Bibliotheken */usr/lib/libm.a* und */usr/lib/libc.so* enthalten?
- Welche Zeichenfolgen sind im Programm *xargs* enthalten?
- Suchen Sie ab dem Verzeichnis "/" alle Dateien, bei denen das *SUID*- oder *Sticky*-Bit gesetzt ist. Zeigen Sie die Dateien mit Ihren Zugriffsrechten an.
- Führen Sie d) auf einem Sparc-Rechner aus. Unterdrücken Sie die Meldung "Permission denied".

1.4 Benutzerverwaltung

- Benutzer muß dem System bekannt gemacht werden, bevor er einen Rechner benutzen kann
- jeder Benutzer besitzt
 - eine Kennung (Nachname, Vorname, Abteilungsname plus Nummer, ...)
 - ein Paßwort
 - ◆ das erste Paßwort wird vom Administrator vergeben
 - ◆ der Benutzer muß sein Paßwort sofort ändern
 - ◆ Paßwörter sollten Groß- und Kleinbuchstaben, Ziffern und Sonderzeichen enthalten
 - ◆ Paßwörter sollten mindestens 8 Zeichen lang sein
 - ◆ Paßwörter dürfen ohne Ziffern und Sonderzeichen in keinem Wörterbuch vorkommen
 - ◆ Paßwörter sollten regelmäßig geändert werden
 - ◆ Paßwörter sollten nicht aufgeschrieben werden
 - ◆ Paßwörter dürfen nicht an andere Personen weitergegeben werden
 - eine Benutzernummer
 - ◆ user identification number (uid)
 - ◆ die Nummern 0-99 sind für das System reserviert
 - eine Gruppennummer
 - ◆ group identification number (gid)
 - ◆ ein Benutzer kann mehreren Gruppen angehören

- ein *Home-Verzeichnis*
 - ◆ i.a. das Verzeichnis */export/home/<Benutzerkennung>*
 - ◆ unter NFS wird dieses Verzeichnis i.a. unter */home/<Benutzerkennung>* zur Verfügung gestellt
 - ◆ ein unbenutztes NFS-Verzeichnis wird i.a. nach einer Wartezeit automatisch aus dem lokalen Dateibaum entfernt
 - ◆ hier befinden sich die Punkt-Dateien für die individuelle Konfigurierung der *Shell* und einiger Anwendungsprogramme (*.cshrc*, *.login*, *.logout*, *.mailrc*, *.emacs*, ...)
 - ◆ der Administrator stellt Standard-Punkt-Dateien beim Einrichten des Benutzers zur Verfügung

- einen Kommandointerpreter (*Shell*)
 - ◆ C-Shell (*csh*)
 - ◆ T-C-Shell (*tcsh*)
 - ◆ Korn-Shell (*ksh*)
 - ◆ Bourne (Again) Shell (*sh*, *bash*)
 - ◆ ...
 - ◆ der Benutzer kann später i.a. einen anderen Kommandointerpreter einstellen
 - ◆ der Benutzer kann jederzeit in einem Kommandointerpreter einen anderen Kommandointerpreter starten

- eine allgemeine Benutzerbeschreibung
(i.a. kann der Benutzer dieses Feld später ebenfalls ändern)

- neben den Benutzerkennungen gibt es i.a. mehrere Systemkennungen
 - die wichtigste Kennung ist *root*
 - die Kennung *root* erlaubt einen uneingeschränkten Zugriff auf den lokalen Rechner
 - weitere Kennungen: siehe */etc/passwd*

- eine Benutzernummer kann für mehrere Benutzerkennungen vergeben werden
 - ⇒ jeder Benutzer kann sich unter seinem Namen mit seinem Paßwort anmelden
 - ⇒ für das System ist es derselbe Benutzer
 - ⇒ sollte nicht gemacht werden, da dann eine eindeutige Zuordnung von Dateibesitzern und Zugriffsrechten nicht möglich ist

- bei großen Systemen können für Teilbereiche sogenannte Unteradministratoren mit eingeschränkten Rechten eingerichtet werden (z.B. Netzwerkadministrator)
 - ⇒ Unteradministratoren müssen bei Solaris der Gruppe 14 (*sysadmin*) angehören

- die Benutzerkennung *lp* wird als Besitzer des Drucksystems geführt
- unter UNIX gibt es viele Pseudobnutzer, denen Dateien gehören (z.B. adm, bin, sys, uucp, ...)
⇒ Pseudobnutzer können sich nicht am System anmelden
- die Benutzerverwaltung erfolgt über verschiedene Dateien
⇒ diese Dateien sollten nur mit den entsprechenden Administrationsprogrammen bearbeitet werden (z.B. useradd, groupadd, passwd, admintool, ...)

- die Paßwortdatei */etc/passwd*
 - enthält alle (lokalen) Benutzerkennungen
 - enthielt früher auch die Paßwörter
(aus Sicherheitsgründen jetzt in der *Shadow*-Datei)

- Zugriffsrechte unter Solaris 7 x86

```
-rw-r--r--  1 root      sys          414 Mär 26 20:24 passwd
```

- Dateiausschnitt

```
root:x:0:1:Super-User:/:/sbin/sh
daemon:x:1:1:/:/
bin:x:2:2:/:usr/bin:
sys:x:3:3:/:/
adm:x:4:4:Admin:/var/adm:
lp:x:71:8:Line Printer Admin:/usr/spool/lp:
uucp:x:5:5:uucp Admin:/usr/lib/uucp:
nuucp:x:9:9:uucp Admin:/var/spool/uucppublic:/usr/lib/uucp/uucico
listen:x:37:4:Network Admin:/usr/net/nls:
nobody:x:60001:60001:Nobody:/:
noaccess:x:60002:60002:No Access User:/:
```

- Bedeutung der Felder

1. Benutzerkennung (root)
2. Paßwortverweis auf *Shadow*-Datei (x)
3. Benutzernummer (0)
4. Gruppennummer (1)
5. ausgeschriebener Benutzername / Kommentar (Super-User)
6. *Home*-Verzeichnis (/)
7. Standard-*Shell* (/sbin/sh)

- die *Shadow-Datei* */etc/shadow*
 - enthält die verschlüsselten Paßwörter
 - Zugriffsrechte unter Solaris 7 x86

```
-r----- 1 root      sys          233 Mär 26 20:24 shadow
```

- Dateiausschnitt

```
root:nTWFVHsO3VsAY:6445::::::  
daemon:NP:6445::::::  
bin:NP:6445::::::  
sys:NP:6445::::::  
adm:NP:6445::::::  
lp:NP:6445::::::  
uucp:NP:6445::::::  
nuucp:NP:6445::::::  
listen:*LK*::::::  
nobody:NP:6445::::::  
noaccess:NP:6445::::::
```

- Bedeutung der Felder
 1. Benutzerkennung (root)
 2. verschlüsseltes Paßwort (nTWFVHsO3VsAY)
 3. letzte Änderung des Paßworts (6445)
 4. Mindestanzahl der Tage, die zwischen Paßwortänderungen verstreichen müssen
 5. Maximalanzahl der Tage, die zwischen Paßwortänderungen verstreichen dürfen
 6. Tage für eine Warnung vor dem Verfall des Paßworts
 7. Anzahl der maximal zulässigen inaktiven Tage der Benutzerkennung
 8. absoluter Tag für den Verfall der Benutzerkennung
 9. wird in Solaris 7 nicht benutzt

- die Angaben zur letzten Änderung des Paßworts und zum Verfallsdatums der Benutzerkennung werden in Tagen seit dem 1.1.1970 angegeben
- anstelle des verschlüsselten Passworts kann NP (no password) stehen (bei Linux: *)
 - ⇒ es handelt sich um einen Pseudobenutzer
- anstelle des verschlüsselten Paßworts kann auch *LK* (locked) stehen
 - ⇒ Benutzerkennung ist augenblicklich gesperrt
- falls das Paßwortfeld leer ist, hat der Benutzer kein Paßwort und kann sich ohne Paßwort beim System anmelden

- die Gruppendatei */etc/group*
 - legt die Gruppen für das System fest
 - Zugriffsrechte unter Solaris 7 x86

```
-rw-r--r--  1 root      sys          278 Mär 26 20:24 group
```

- Dateiausschnitt

```
root::0:root
other::1:
bin::2:root,bin,daemon
sys::3:root,bin,sys,adm
adm::4:root,adm,daemon
uucp::5:root,uucp
mail::6:root
tty::7:root,tty,adm
lp::8:root,lp,adm
nuucp::9:root,nuucp
staff::10:
daemon::12:root,daemon
sysadmin::14:
nobody::60001:
noaccess::60002:
nogroup::65534:
```

- Bedeutung der Felder
 1. Gruppenname (root)
 2. Gruppen-Paßwort
 3. Gruppennummer (0)
 4. Benutzer, der in dieser Gruppe Sekundärmitglied ist
- falls ein Gruppen-Paßwort vergeben wurde, kann auch ein Benutzer in die Gruppe wechseln, der nicht Gruppenmitglied ist, aber das Paßwort kennt
- ein Benutzer kann Mitglied in mehreren Sekundärgruppen sein, aber nur Mitglied in einer Primärgruppe (Gruppe in */etc/passwd*)

- Befehle zur Benutzerverwaltung
 - useradd
 - groupadd
 - passwd
 - über graphische Oberfläche: admintool (bei Linux: yast)

- Eindeutigkeit von Benutzer- und Gruppennummern
 - falls ein Benutzer oder eine Gruppe aus den Systemdateien gelöscht wird, kann es noch Dateien geben, die dem Benutzer / der Gruppe zugeordnet sind
 - ⇒ das System kann die Nummer nicht mehr durch einen Namen auflösen
 - ⇒ *ls -l* würde z.B. folgendes liefern:

```
-rw-r--r--  1 1099      stud           278 Mär 26 20:24 test
```

(statt z.B. fd1099 wird nur noch 1099 ausgegeben)
 - kann zu Sicherheitsproblemen führen, wenn die Nummer später einem neuen Benutzer zugeteilt wird
 - ⇒ wird automatisch Eigentümer aller Dateien

- unter NFS sollten alle Benutzer auf lokalen Rechnern dieselben Benutzer- und Gruppennummern haben wie auf den *Servern*
 - ⇒ andernfalls könnte ein Benutzer A auf dem *Server* eine Datei ablegen, die nur mit seiner lokalen Benutzernummer angelegt wird, da der *Server* die lokale Nummer nicht in den Benutzer-
namen auflösen kann
 - ⇒ greift jetzt ein Benutzer B von einem anderen lokalen Rechner mit derselben lokalen Benutzernummer auf die Datei des *Servers* zu, gilt er als Eigentümer der Datei !
 - ⇒ **eine netzwerkweit eindeutige Vergabe von Benutzer- und Gruppennummern ist zwingend erforderlich**

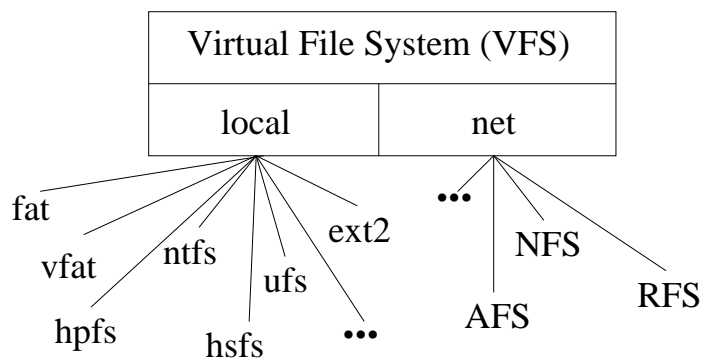
Aufgabe 1-3:

Führen Sie die folgenden Aufgaben jeweils unter Solaris und Linux aus.

- a) Legen Sie mit Hilfe des Befehls *groupadd* die neue Gruppe *stud* an.
- b) Legen Sie mit Hilfe des Befehls *useradd* einen neuen Benutzer an. Vergeben Sie für den neuen Benutzer ein Anfangspasswort.
- c) Legen Sie über die graphische Oberfläche eine neue Gruppe *projekt* an.
- d) Legen Sie über die graphische Oberfläche einen neuen Benutzer in der Primärgruppe *stud* und der Sekundärgruppe *projekt* an. Vergeben Sie für den neuen Benutzer ein Anfangspasswort. Nutzen Sie alle Möglichkeiten des Systems für sichere Passwörter aus (Änderung: alle 4 Wochen, Verfallswarnung: 10 Tage, ...)

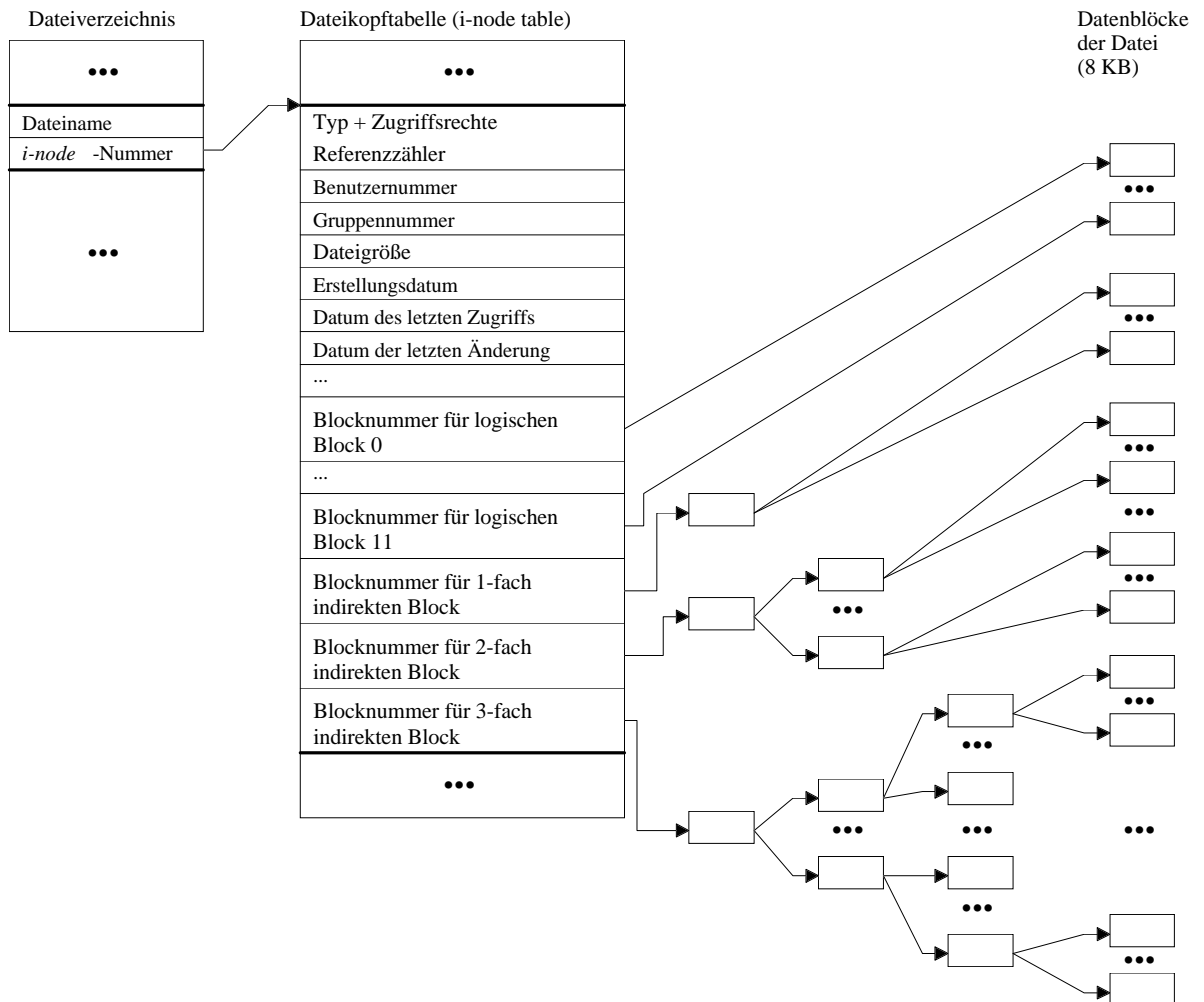
1.5 Dateiverwaltung

- hierarchisches Dateisystem
(Startpunkt: /, root directory, Wurzelverzeichnis, Wurzelkatalog, ...)
- transparentes Dateisystem
 - unterschiedliche Dateisysteme sehen für den Benutzer gleich aus
 - Benutzer kann nicht unterscheiden, ob das Dateisystem lokal oder über ein Rechnernetz zur Verfügung gestellt wird
 - Realisierung über ein virtuelles Dateisystem (*virtual file system*)



- Dateisysteme können automatisch oder manuell eingehängt bzw. abgehängt werden
- der Aufbau einiger Dateisysteme kann der Folie zum "Layout einer PC-Festplatte" entnommen werden (Folie 1-5)

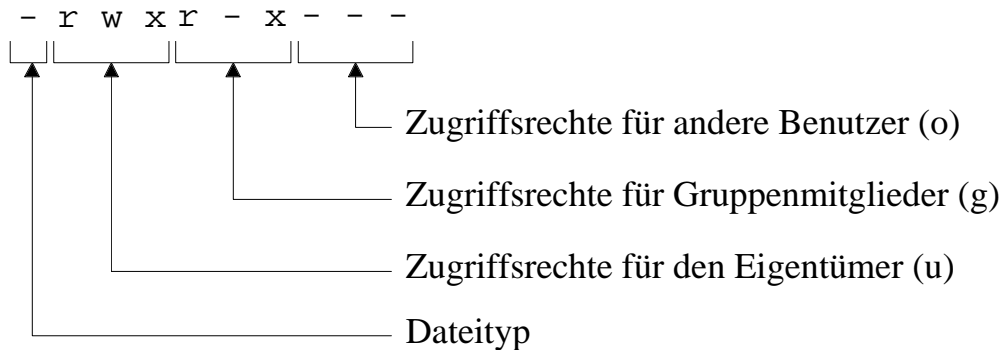
- Verweisstruktur im *ufs*-Dateisystem zur Organisation des Zugriffs auf die Datenblöcke einer Datei



- Geräte werden syntaktisch wie Dateien angesprochen
- Dateisystem kann zur Interprozesskommunikation genutzt werden (*pipe*)

1.5.1 Dateitypen

- Aufbau des Dateityp- / Zugriffsrechtefeldes



- Dateitypen

- normale Dateien
(Textdateien, *Shell*-Skripte, Objektdateien, ausführbare Programme, ...)
- b blockorientierte Gerätedatei
- c zeichenorientierte Gerätedatei
- d Dateiverzeichnis
- D Verweisdatei für Kommunikation (door)
- l symbolischer Verweis (*symbolic link*; Kommando `ln -s`)
- p FIFO-Datei (*named pipe*)
- s *AF_Unix Socket*-Datei für Prozeßkommunikation

- Dateityp wird anhand einer Bytefolge am Anfang der Datei erkannt (*magic number*)
- bekannte Bytefolgen sind in der Datei `/etc/magic` gespeichert
- der Dateityp kann mit Hilfe des Kommandos `file` bestimmt werden
(Es kann sich auch dann um eine Datei im Binärformat handeln, wenn `file` eine Text- oder Datendatei meldet. Das Kommando `file` benutzt `/etc/magic` zur Bestimmung des Dateityps.)

- Zugriffsrechte
 - 3-mal drei Bits
(erste Bit-Gruppe für Eigentümer, zweite für Gruppenmitglieder, dritte für alle anderen Benutzer)
 - 1. Bit einer Gruppe: Lesen erlaubt / nicht erlaubt
 - 2. Bit einer Gruppe: Schreiben / Löschen erlaubt / nicht erlaubt
 - 3. Bit einer Gruppe: Ausführen der Datei erlaubt / nicht erlaubt
 - Codierung der Werte
 - kein Recht
 - r Leserecht (read)
 - w Schreib-/Löschrecht (write)
 - x ausführbar (execute)
 - s *SUID/SGID*-Bit und *execute* gesetzt
 - S / l / L *SUID/SGID*-Bit gesetzt
 - t *Sticky*-Bit und *execute* gesetzt
 - T *Sticky*-Bit gesetzt
 - falls das *SUID*-Bit gesetzt ist, hat das Programm während der Ausführung die Rechte des Besitzers
 - falls das *SGID*-Bit gesetzt ist, hat das Programm während der Ausführung die Rechte der Gruppe
 - damit ein beliebiger Benutzer Programme mit gesetztem *SUID* / *SGID*-Bit ausführen kann, müssen zusätzlich Ausführungsrechte für *andere Benutzer* gesetzt sein (3. Bit-Gruppe)

- falls Ausführungsrechte für den Besitzer / die Gruppe gesetzt sind, wird "s" benutzt
- falls keine Ausführungsrechte für den Besitzer / die Gruppe gesetzt sind, wird S, l oder L benutzt
(in diesem Fall haben die Bits keine Bedeutung; sie haben ebenfalls keine Bedeutung bei Verzeichnissen)
- Programme mit gesetztem *SUID* / *SGID*-Bit können eine Sicherheitslücke sein
- **auf keinen Fall bei *Shell*-Skripten das *SUID* / *SGID*-Bit setzen**
- bei Solaris stellt das *SGID*-Bit bei Datendateien die exklusive Nutzung der Datei sicher (mandatory file locking)
- bei ausführbaren Programmen sorgt das *Sticky*-Bit (save text image after execution) dafür, daß das Programm auch dann im Hauptspeicher (*Swap*-Bereich) bleibt, wenn es augenblicklich von niemandem benutzt wird (ist heute ohne Bedeutung)
- bei Dateiverzeichnissen sorgt das *Sticky*-Bit dafür, daß nur der Besitzer einer Datei die Datei löschen kann
(Beim Verzeichnis */tmp* haben alle Benutzer Schreib- und damit Löschrechte im Verzeichnis. Damit könnte jeder Benutzer jede Datei in diesem Verzeichnis löschen (auch Dateien des Benutzers *root*, die keine Rechte für irgendeine Gruppe oder alle anderen Benutzer haben). Durch das *Sticky*-Bit kann dies verhindert werden.)

1.5.2 Kommandos

- das Kommando *ls*

Option	Bedeutung
a	alle Dateien anzeigen (auch versteckte Dateien)
d	Unterverzeichnis nur mit eigenem Eintrag anzeigen
i	für jede Datei die <i>i-node</i> -Nummer ausgeben
l	langes Ausgabeformat
R	rekursive Auflistung der Dateien / Verzeichnisse
t	Sortierreihenfolge nach Veränderungszeitpunkt (normalerweise wird alphabetisch sortiert)

– Bedeutung der Felder

```
kea:/usr/bin # ls -ail yp*
12869 -rwxr-xr-x  1 root  root    4000 Jul 24  1998 ypcat
12871 -rwxr-xr-x  3 root  root   10476 Jul 24  1998 ypchfn
12871 -rwxr-xr-x  3 root  root   10476 Jul 24  1998 ypchsh
12871 -rwxr-xr-x  3 root  root   10476 Jul 24  1998 yppasswd
...
```

1. *i-node*-Nummer
2. Dateityp und Zugriffsberechtigung
3. Anzahl der Verweise auf die Datei;
bei Verzeichnissen Anzahl der Unterverzeichnisse (s. unten)
4. Dateibesitzer
5. Gruppenzugehörigkeit
6. Dateigröße in Byte
7. Erstellungsdatum
8. Erstellungsuhrzeit oder -jahr
9. Dateiname

```
kea:/usr/lib # ls -dli yp
224928 drwxr-xr-x  3 root  root    1024 Dec  1  1998 yp
```

```
kea:/usr/lib # ls -ail yp
224928 drwxr-xr-x  3 root  root    1024 Dec  1  1998 .
61201 drwxr-xr-x 43 root  root    6144 Nov 14 09:05 ..
204513 drwxr-xr-x  2 root  root    1024 Aug 26  1998 ypmake
...
```

- das Kommando *chgrp [Optionen] gruppe Dateien*

Option	Bedeutung
-h	die Gruppe einer <i>Link</i> -Datei ändern (normalerweise wird nur die Gruppe der Datei, auf die der <i>Link</i> zeigt, geändert)
-R	rekursiv die Gruppe aller Dateien ändern

- das Kommando *chown [Optionen] name Dateien*

Option	Bedeutung
-h	den Eigentümer einer <i>Link</i> -Datei ändern (normalerweise wird nur der Eigentümer der Datei, auf die der <i>Link</i> zeigt, geändert)
-R	rekursiv den Eigentümer aller Dateien ändern

mit *chown [Optionen] name:gruppe Dateien* kann der Eigentümer und die Gruppenzugehörigkeit der Datei geändert werden

- das Kommando *chmod [Option] modus Dateien*

Option	Bedeutung
-R	rekursiv den Eigentümer aller Dateien ändern

- der Zugriffsmodus kann als Oktalzahl oder symbolisch angegeben werden

- die Oktalzahl ergibt sich aus der Addition folgender Werte

4000 Benutzernummer des Dateibesitzers als effektive Benutzernummer verwenden

20x0 Gruppennummer des Dateibesitzers als effektive Gruppennummer verwenden, falls x ungerade ist. Sonst wird eine exklusive Benutzung der Datei sichergestellt.

1000 *Sticky*-Bit setzen

400 Lesezugriff für den Besitzer

200 Schreibzugriff für den Besitzer

100 Ausführungsrecht für den Besitzer einer Datei.
Zugriff auf ein Verzeichnis für den Besitzer.

40 Lesezugriff für die Gruppe

20 Schreibzugriff für die Gruppe

10 Ausführungsrecht oder Verzeichniszugriff für die Gruppe

4 Lesezugriff für andere Benutzer

2 Schreibzugriff für andere Benutzer

1 Ausführungsrecht oder Verzeichniszugriff für andere Benutzer

- Format für symbolische Modusangabe: <Kennzeichen><Recht>

<Kennzeichen> u Eigentümer (login **u**ser)

g Gruppe (**g**roup)

o alle anderen Benutzer (**o**ther user)

a alle (Wert für alle drei Bit-Gruppen)

<Recht>	+<r> Recht hinzufügen
	-<r> Recht entziehen
	=<r> lösche alle Rechte außer
<r>	r Leserecht
	w Schreib- / Löschrecht
	x Ausführungsrecht für Dateien
	Suchrecht in Verzeichnissen
	s SUID- /SGID-Bit setzen
	l SGID-Bit setzen (Solaris)
	t Sticky-Bit setzen

- eine Datei kann gelöscht werden, wenn eine Schreiberlaubnis auf das Verzeichnis existiert

- Beispiele:

```
chmod -R 755 /usr/bin/*
```

```
chmod 7777 xyz
```

```
chmod a=r *
```

```
chmod -R g+w *
```

```
chmod -R g+w,o=r *
```

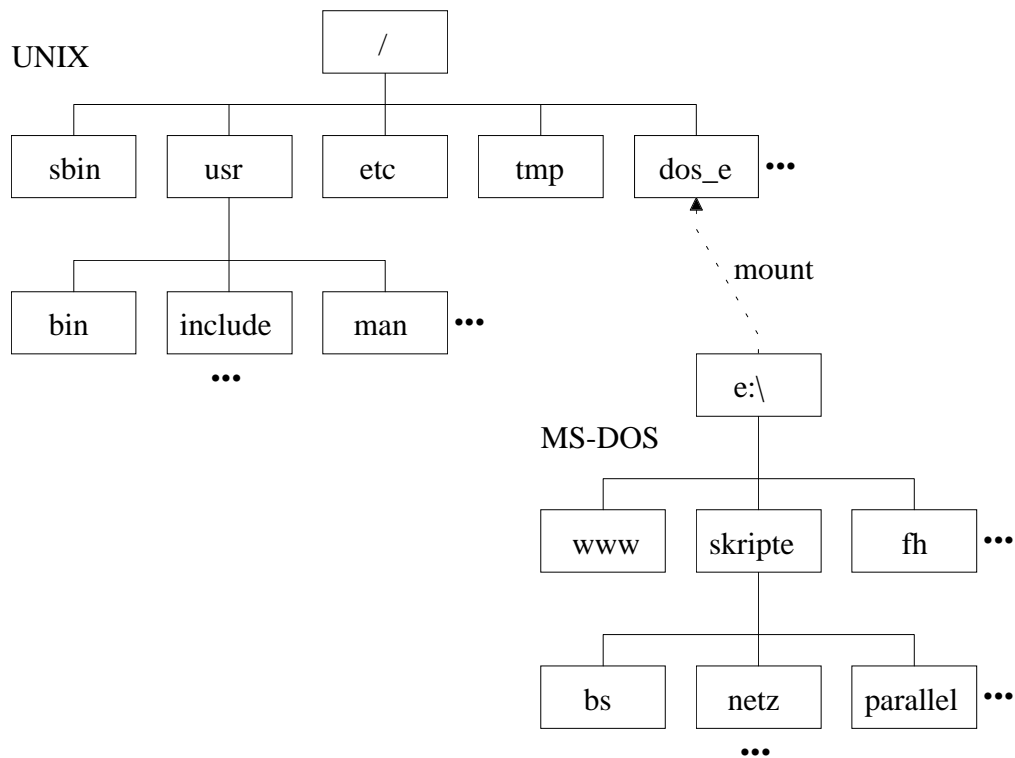
- das Kommando *umask* dient zur Voreinstellung von Rechten (im Gegensatz zu *chmod* gibt es an, welche Rechte **nicht** erlaubt sein sollen)

chmod 644 ... Lese-/Schreibrecht für Besitzer, Leserecht für Gruppe und andere Benutzer

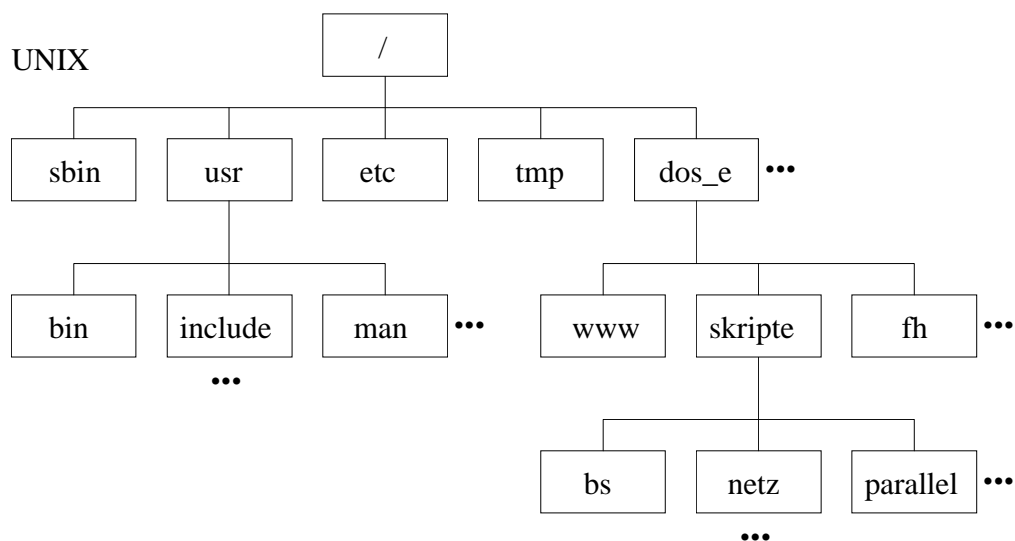
umask 022 hat auf Dateien dieselbe Wirkung (bei Verzeichnissen wäre auch das *execute*-Bit gesetzt)

1.5.3 Einhängen / Abhängen von Dateisystemen

- vor Ausführung des Befehls *mount*



- nach Ausführung des Befehls *mount*



1.5.4 Einhängen von Dateisystemen beim Systemstart

- Linux: `/etc/fstab`

```
# device mount-      fs-type options                                dump fsck-
#         point                                     order

/dev/sdb7 swap          swap    defaults                                0    0
/dev/sda5 /                ext2    defaults                                1    1
/dev/sdb6 /usr          ext2    defaults                                1    2
/dev/sdb8 /export/home ext2    defaults                                1    2
/dev/scd0 /cdrom       iso9660 ro,noauto,user                        0    0
/dev/fd0  /fd0         msdos   rw,noauto,user                        0    0
none     /proc        proc    defaults                                0    0
```

- Solaris: `/etc/vfstab`

```
#device          device          mount          FS    fsck mount  mount
#to mount       to fsck        point         type  pass at boot opt.
#
#/dev/dsk/c1d0s2 /dev/rdisk/c1d0s2 /usr          ufs   1    yes   -
fd              -              /dev/fd       fd    -    no    -
/proc           -              /proc         proc  -    no    -
/dev/dsk/c0t1d0s1 -              -             swap  -    no    -
/dev/dsk/c0t0d0s0 /dev/rdisk/c0t0d0s0 /             ufs   1    no    -
/dev/dsk/c0t0d0s6 /dev/rdisk/c0t0d0s6 /usr          ufs   1    no    -
/dev/dsk/c0t1d0s3 /dev/rdisk/c0t1d0s3 /var          ufs   1    no    -
/dev/dsk/c0t1d0s7 /dev/rdisk/c0t1d0s7 /export/home  ufs   2    yes   -
/dev/dsk/c0t0d0s5 /dev/rdisk/c0t0d0s5 /opt          ufs   2    yes   -
swap           -              /tmp          tmpfs -    yes   -
```

- `/etc/mnttab` gibt an, welche Dateisysteme aktuell eingebunden sind

Aufgabe 1-4:

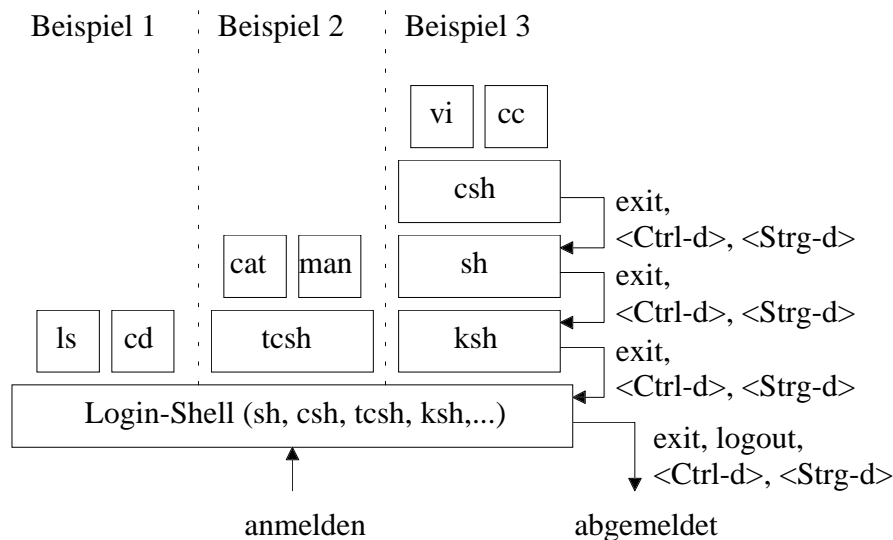
Führen Sie die folgenden Aufgaben jeweils unter Solaris und Linux aus.

- a) Erzeugen Sie das Verzeichnis `"/dosc"` und hängen Sie in dieses Verzeichnis manuell die MS-DOS-Partition ein. Kopieren Sie eine Datei auf die MS-DOS-Partition. Was passiert mit langen Namen? Können Sie die Datei bearbeiten?
- b) Erweitern Sie die Datei `"/etc/(v)fstab"`, so daß die MS-DOS-Partition automatisch beim Starten des Betriebssystems in das Verzeichnis `"/dosc"` eingehängt wird. Setzen Sie geeignete Optionen, damit MS-DOS-Programme unter UNIX nicht ausführbar sind und auf gar keinen Fall mit gesetztem SUID-Bit ausgeführt werden können usw..
- c) Legen Sie eine kleine Dateiverzeichnisstruktur an, in der neben Unterverzeichnissen auch *Hard-* und *Symbolic-Links* existieren. Ändern Sie die Eigentümer-/Gruppenzugehörigkeit **aller** Dateien in einem Schritt, so daß die Dateien danach einem normalen Benutzer gehören.
- d) Ändern Sie die Zugriffsrechte der Dateien aus c), so daß Benutzer der Gruppe "projekt" mit den Dateien genauso arbeiten können wie der Eigentümer.

1.6 *Shell*-Programmierung

- *Shell*-Programme sind die klassische Schnittstelle zwischen Benutzer und Betriebssystem
- *Shell* ist ein normales Programm
- in der *Shell* können UNIX-Kommandos eingegeben werden
- *Shell* erlaubt Ein-/Ausgabeumleitung und Fließbandverarbeitung (pipeline)
- Befehlseingaben werden immer von links nach rechts interpretiert
- eine *Shell* ist sowohl ein Kommando-Interpreter als auch eine Art Programmiersprache
- Kommandos können in einer Textdatei (*Shell*-Skript) abgelegt und dann ausgeführt werden
(der Datei müssen vorher mit *chmod* Ausführungsrechte zugewiesen werden)
- die erste Zeile einer *Shell*-Skriptdatei enthält i.a. einen Hinweis auf die *Shell*, für die die Datei geschrieben wurde
 - `#!/bin/sh` *Bourne-Shell*-Skript
 - `#!/bin/csh -f` *C-Shell*-Skript
 - `#!/bin/ksh` *Korn-Shell*-Skript
 - usw.

- jeder Benutzer erhält eine *Shell* beim *Login* als *Login-Shell* zugeteilt (ist in der Datei */etc/passwd* bzw. der entsprechenden NIS/NIS+-Tabelle festgelegt; kann i.a. mit dem Kommando *passwd* geändert werden)
- in einer *Shell* können beliebige weitere *Shells* aufgerufen werden



- für jede *Shell* gibt es systemweite und/oder benutzerspezifische Initialisierungsdateien

- */etc/profile* *Bourne-Shell*
~/.profile
- */etc/.login* *C-Shell*
~/.cshrc
~/.login
~/.logout
- */etc/csh.cshrc* oder */etc/.cshrc* *T-C-Shell*
/etc/csh.login oder */etc/login* oder */etc/.login*
~/.tcshrc oder *~/.cshrc*
~/.login
/etc/csh.logout oder */etc/logout*
~/.logout

- die Initialisierungsdateien werden i.a. in der angegebenen Reihenfolge ausgeführt, falls sie existieren
(„oder“-verknüpfte Dateien werden von links nach rechts bis zur ersten Fundstelle gesucht)
- es werden nicht alle Dateien von allen Betriebssystemen unterstützt (manche Betriebssysteme benutzen auch noch andere Namen)
- *login*- und *logout*-Skripte werden nur von der *Login-Shell* ausgeführt
- alle anderen Skriptdateien werden von jeder *Shell* ausgeführt
- unter Solaris sind standardmäßig die *Shells* *csh*, *ksh*, *sh* und *jsh* erlaubt
(falls man Benutzern und Programmen andere *Shells* erlauben will, muß die Datei */etc/shells* erzeugt werden, in die die Pfade aller erlaubten *Shells* eingetragen werden)
- die Rechte eines Benutzers können unter Solaris durch die *Restricted-Bourne-Shell* (*/usr/lib/rsh*) eingeschränkt werden
(diese *Shell* darf nicht mit der *Remote Shell* (*/usr/bin/rsh*) verwechselt werden!)
 - ◆ der Benutzer kann sein Verzeichnis nicht in hierarchisch höhere Verzeichnisse verlassen
 - ◆ er darf die Umgebungsvariablen *SHELL* und *PATH* nicht verändern
 - ◆ er darf keine Kommandos mit Pfadangaben ausführen
 - ◆ die *rsh* bietet keine Erhöhung der Systemsicherheit, wenn der Benutzer Schreibrechte in seinem Verzeichnis hat
(er kann dann ein *Shell*-Skript schreiben, in dem eine normale *Shell* gestartet wird)

- falls eine *Shell* durch eine andere *Shell* überlagert wird, werden alle exportierten Variablen (Umgebungsvariablen) an die darüberliegende *Shell* weitergegeben

- Umgebungsvariablen können mit *env* bzw. *printenv* angezeigt werden
 - DISPLAY der zu benutzende Bildschirm, z.B. *tyr:0.0*
 - EDITOR der Standard-Editor, z.B. *xemacs*
 - GROUP die Benutzergruppe, z.B. *stud*
 - HOME Pfad zum *HOME*-Verzeichnis, z.B. */home/gross*
 - HOST Rechnername, z.B. *tyr*, *tyr.informatik.fh-fulda.de*
 - IFS *Internal Field Separator*, i.a. Leer-/Tabulatorzeichen und *<newline>*
 - LD_LIBRARY_PATH Pfade zu Bibliotheken
 - LOGNAME Benutzername, z.B. *gross*
 - MANPATH Pfade zu Handbuchseiten
 - PATH Pfade zu Programmen
 - PS1 primäre Eingabeaufforderung (i.a. \$, %, #)
 - PS2 sekundäre Eingabeaufforderung (i.a. >)
 - SHELL Name der *Login-Shell*
 - TERM Typ des Sichtgeräts, z.B. *xterm*, *vt100*
 - USER Benutzername, z.B. *gross*
 - und viele weitere

- jede *Shell* stellt Befehle zur Verfügung, die in der *Shell* implementiert sind
- *Shell*-Befehle sind für die direkte Arbeit mit der *Shell* notwendig
- für *Shell*-Befehle werden keine eigenen Prozesse erzeugt
- eine *Shell* analysiert eine gelesene Eingabe, expandiert ggf. die Parameter nach ihren Regeln und führt dann das Kommando aus (als *Shell*-Kommando oder als eigenständiges Programm)
- Apostroph oder Anführungszeichen
 - Zeichenfolgen mit Leerzeichen müssen in Apostroph oder Anführungszeichen eingeschlossen werden
 - steuern die Art der Interpretation einer Zeichenfolge
 - Anführungszeichen erlauben eine Parametersubstitution in der Zeichenfolge
(Beispiel: `echo "Benutzername: $USER" ⇒ Benutzername: gross`)
 - das Apostroph verbietet eine Parametersubstitution
(Beispiel: `echo 'Benutzername: $USER' ⇒ Benutzername: $USER`)
- Accent grave (*backquote*)
 - die Zeichenfolge wird zuerst durch die *Shell* ausgewertet und das Ergebnis wird dann substituiert
 - wird i.a. verwendet, um ein Kommando auszuführen und das Ergebnis des Kommandos zu benutzen
(Beispiel: `setenv SYSTEM_ENV `uname -s``)

- *Shell*-Variablen

Variable	Bedeutung	sh	csch
\$#	Anzahl Parameter	x	
\$-	Optionen mit denen die <i>Shell</i> aufgerufen wurde	x	
\$?	Rückgabewert des zuletzt ausgeführten Kommandos	x	
\$\$	Prozeßnummer des aktuellen Prozesses	x	x
#!	Prozeßnummer des letzten Hintergrundprozesses	x	
\$n	<i>n</i> -ter Parameter, $1 \leq n \leq 9$, Nummerierung von links nach rechts	x	x
\$0	Name der aktuellen <i>Shell</i> bzw. des aktuellen Programms	x	x
\$*	alle Parameter ("\$1 \$2 ... \$9")	x	x
@	alle Parameter mit eigenen Anführungszeichen ("\$1" "\$2" ... "\$9")	x	
\$argv[n]	<i>n</i> -ter Parameter		x
\${argv[n]}	<i>n</i> -ter Parameter		x
#argv	Anzahl Parameter		x
\$var	Wert der Variablen <i>var</i>	x	x
\${var}	Wert der Variablen <i>var</i>	x	x
\$?var	1, falls <i>var</i> gesetzt ist; 0 sonst		x
\${?var}	1, falls <i>var</i> gesetzt ist; 0 sonst		x
\$<	liest Zeile von <i>stdin</i>		x

Variable	Bedeutung	sh	csH
<code>\${var:-wert}</code>	falls <i>var</i> definiert ist und einen Wert besitzt, wird der Wert von <i>var</i> substituiert; andernfalls wird der Wert <i>wert</i> substituiert	x	
<code>\${var:=wert}</code>	falls <i>var</i> nicht definiert ist oder noch keinen Wert besitzt, wird <i>var</i> der Wert <i>wert</i> zugewiesen; anschließend wird der Wert von <i>var</i> substituiert; Positionsparameter (<i>\$1</i> , ...) können auf diese Weise nicht initialisiert werden	x	
<code>\${var:?wert}</code>	falls <i>var</i> definiert ist und bereits einen Wert besitzt, wird dieser Wert substituiert; andernfalls wird <i>wert</i> als Fehlermeldung ausgegeben und das Skript abgebrochen; falls <i>wert</i> weggelassen wird, wird die Meldung "parameter null or not set" ausgegeben	x	
<code>\${var:+wert}</code>	falls <i>var</i> definiert ist und einen Wert besitzt, wird der Wert <i>wert</i> substituiert; andernfalls wird eine leere Zeichenfolge substituiert	x	

- Ein-/Ausgabeumleitung

- Dateideskriptoren

0	stdin
1	stdout
2	stderr

- Umleitungsoperationen

Symbol		sh	csch
>	Ausgabe umleiten	x	x
>>	Ausgabe am Ende der Datei anfügen	x	x
<	Eingabe umleiten	x	x
	Fließbandverarbeitung (<i>pipe</i>)	x	x
>& file	<i>stdout</i> und <i>stderr</i> in Datei umleiten		x
>>& file	<i>stdout</i> und <i>stderr</i> an Datei anfügen		x
& cmd	<i>stdout</i> und <i>stderr</i> an <i>cmd</i> weiterleiten		x
2> file	<i>stderr</i> in Datei umleiten	x	
> file 2>&1	<i>stdout</i> und <i>stderr</i> in Datei umleiten	x	
>>file 2>&1	<i>stdout</i> und <i>stderr</i> an Datei anhängen	x	
2>&1 cmd	<i>stdout</i> und <i>stderr</i> an <i>cmd</i> weiterleiten	x	

- *stdout* und *stderr* in verschiedene Dateien umleiten

csch: (Kommando > Ausgabedatei) >& Fehlerdatei

sh: Kommando 1> Ausgabedatei 2> Fehlerdatei oder
Kommando > Ausgabedatei 2> Fehlerdatei

- Beispiel:

csch: make |& tee make.log

sh: configure 2>&1 | tee configure.log

- weitere spezielle Symbole für die *sh* und *csh*
 - ; Kommando-Trennzeichen (sequentielle Abarbeitung)
 - & Prozeß im Hintergrund ausführen (\Rightarrow asynchron, parallel)
 - && das folgende Kommando ausführen, wenn das vorhergehende erfolgreich war
 - `cc -c xyz.c && ld -o xyz xyz.o -lc`
 - || das folgende Kommando ausführen, wenn das vorhergehende nicht erfolgreich war
 - `ls xyz || echo "xyz existiert nicht"`
 - () Kommandos innerhalb der Klammer in einer Sub-Shell ausführen; die Ausgabe der Sub-Shell kann wie oben behandelt werden
 - \ Fluchtsymbol (*escape*), d.h. das folgende Zeichen soll von der Shell nicht interpretiert werden
 - # alle folgenden Zeichen bis *<newline>* sind Kommentar
 - ~ HOME-Verzeichnis des Benutzers in Pfadangaben, z.B.
 - `~/login` (gibt es nur bei *csh*)
 - ~xyz LOGIN-Verzeichnis des Benutzers xyz in Pfadangaben, z.B.
 - `~xyz/login` (gibt es nur bei *csh*)
 - @ Werte von Variablen setzen / anzeigen (gibt es nur bei *csh*; entspricht in etwa *expr* in der Bourne-Shell)
 - ...
- in den beiden folgenden Unterkapiteln werden einige Eigenschaften und Kommandos der Bourne- und der C-Shell vorgestellt
(eine vollständige Beschreibung liefern die Handbuchseiten bzw. die Online-Dokumentation: *man csh, man sh, ...*)

1.6.1 Bourne-Shell

- entwickelt von *Steve Bourne*
- älteste *Shell*
- steht unter */usr/bin/sh* als dynamisch gebundene *Shell* und unter */sbin/sh* als statisch gebundene *Shell* zur Verfügung
(*root* sollte als *Login-Shell* immer */sbin/sh* benutzen, da er sich bei dynamisch gebundenen *Shells* bei Systemproblemen u.U. nicht anmelden könnte)
- die *Bourne-Shell* beendet i.a. alle Hintergrundprozesse eines Benutzers beim *Logout*
(Ausnahme: der Prozeß wurde mit dem Kommando *nohup* gestartet)
- *Shell*-Skripte für die *Bourne-Shell* laufen bis auf einige Spezialfälle auch unter der *Korn-Shell* (eine Weiterentwicklung der *Bourne-Shell*)
- *Shell*-Variablen setzen / löschen
 - `name=wert` (vor und nach dem Gleichheitszeichen darf kein Leerzeichen stehen)
 - **Beispiel:** `xyz="Alles klar?"`
`echo $xyz` ⇒ Alles klar?
`sh`
`echo $xyz` ⇒

⇒ Variable ist nur lokal (in der aktuellen *Shell*) bekannt

- mit *export* wird aus der lokalen Variablen eine globale Umgebungsvariable

```
xyz=klar?
```

```
export abc xyz (noch nicht existierende Variablen können exportiert werden)
```

```
abc=Alles
```

Welche Ausgabe liefert: (export; env; set) | egrep "abc|xyz"

```
sh
```

```
echo "$abc $xyz" ⇒ Alles klar?
```

```
abc=Na; xyz=klar! (diese Änderung ist nur lokal!)
```

```
echo "$abc $xyz" ⇒ Na klar!
```

Welche Ausgabe liefert: (export; env; set) | egrep "abc|xyz"

```
sh
```

```
echo "$abc $xyz" ⇒ Alles klar?
```

```
exit; exit
```

```
echo "$abc $xyz" ⇒ Alles klar?
```

```
abc=Na; xyz=klar! (diese Änderung ist wieder global!)
```

Welche Ausgabe liefert: (export; env; set) | egrep "abc|xyz"

```
sh
```

```
echo "$abc $xyz" ⇒ Na klar!
```

⇒ wenn ein Sohnprozeß eine exportierte Variable des Vaterprozesses ändert und selbst exportiert, sehen nur seine Sohnprozesse die Änderungen (die Variable des Vaterprozesses behält ihren ursprünglichen Wert)

- mit *unset* kann eine Variable gelöscht werden, z.B. *unset abc*

- mit *readonly* kann eine Konstante definiert werden, z.B.

```
xyz=ok
```

```
readonly xyz (von jetzt an kann xyz kein Wert zugewiesen werden)
```

Konstanten können mit *unset* nicht entfernt werden!

- lokale Variablen überdecken globale Variablen gleichen Namens

- Klammerung von Kommandos
 - (Kommandoliste)
 - ⇒ Kommandoliste in einer Sub-*Shell* ausführen
 - { Kommandoliste; }
 - ⇒ Kommandoliste in der aktuellen *Shell* ausführen
 - ⇒ das Semikolon und die Leerzeichen sind erforderlich
 - funktionsname() { Kommandoliste; }
 - ◆ definiert eine Funktion, die über *funktionsname* aufgerufen werden kann
 - ◆ das Semikolon und die Leerzeichen sind erforderlich
 - ◆ **Beispiel:** dir() { ls -al \$@; }

- Eingabe in ein *Shell*-Skript

```
#!/bin/sh
echo "Eingabe: \c" # \c verhindert <newline>
read eingabe
echo "eingegebene Zeichenfolge: $eingabe"
```

- bedingte Ausführung

```
if Bedingung_1; then
    cmd_liste_1
[elif Bedingung_2; then
    ...]
[else
    cmd_liste_n]
fi
```

- das Semikolon kann entfallen, wenn *then* auf einer eigenen Zeile steht
- optional können mehrere *elif*-Anweisungen folgen
- optional kann genau eine *else*-Anweisung folgen
- die Bedingung wird mit der *test*-Anweisung oder dem `[]`-Operator ausgewertet (nach "[" und vor "]" **muß** ein Leerzeichen stehen!)
Beispiel: `if [$# -ge 2]; then`
`if test $# -ge 2; then`
- *test* / `[]` erlaubt u.a. die folgenden Operatoren
 - ♦ Vergleich zweier Zahlen: `-eq`, `-ne`, `-gt`, `-ge`, `-lt`, `-le`
 - ♦ logische Operationen: `!` (Negation), `-a` (binäres und), `-o` (binäres oder), `()` (Gruppierung; damit die Klammern von der *Shell* nicht interpretiert werden, muß `"\"` und `"\"` benutzt werden)
Beispiel: `if [\"($# -eq 2 \") -a \"($1 -lt 10 \")]; then`

- ◆ Vergleiche mit Zeichenfolgen:

-z str	wahr, falls Länge der Zeichenfolge gleich 0 abc=""; test -z "\$abc" && echo gleich_null
-n str	wahr, falls Länge der Zeichenfolge ungleich 0
str1 = str2	str1 und str2 sind gleich
str1 != str2	str1 und str2 sind ungleich
str	wahr, falls Zeichenfolge leer ist abc=; test "\$abc" echo NULL

- ◆ Operationen mit Dateien

-r	wahr: Datei existiert und <i>lesen</i> erlaubt
-x	wahr: Datei existiert und <i>ausführen</i> erlaubt
-d	wahr: Datei existiert und ist Verzeichnis
-s	wahr: Datei existiert, Dateigröße ist größer 0
...	

- Verzweigung

```
case wort in
  muster1) cmd_liste_1;;
  muster2) cmd_liste_2;;
  ...
  *)      cmd_liste_n;;
esac
```

- *wort* wird in der Reihenfolge der Anweisungen mit den Mustern verglichen
- bei Übereinstimmung wird die entsprechende *cmd_liste* ausgeführt

- jede Alternative muß mit ";;" abgeschlossen werden
 - ";;" kann hinter der letzten Anweisung einer Kommandoliste oder auf einer neuen Zeile stehen
 - "*" trifft auf jedes *wort* zu, falls *wort* mit keinem Muster übereinstimmt (entspricht *default*-Anweisung)
 - eine Kommandoliste kann auch für mehrere oder-verknüpfte Muster zutreffen
muster1a|muster1b|...|muster1n) ...;;
 - Muster dürfen die Zeichen "?" (ein beliebiges Zeichen) und "*" (beliebige Anzahl Zeichen) enthalten
 - sobald eine Kommandoliste abgearbeitet wurde, werden die restlichen Alternativen nicht mehr überprüft
- *for*-Schleife

```
for var [in Werteliste] do
    cmd_liste
done
```
 - falls die Werteliste fehlt, wird "\$@" benutzt (\$# Werte)
 - "var" durchläuft alle Werte bzw. Positionsparameter
 - analog zur Sprache C kann *continue* bzw. *break* benutzt werden, um die aktuelle Iteration bzw. die gesamte Schleife zu beenden
 - die Werteliste kann mit Hilfe von "?" und "*" gebildet werden

- *while*-Schleife

```
while Bedingung do
  cmd_liste
done
```

analog zur Sprache C kann *continue* bzw. *break* benutzt werden, um die aktuelle Iteration bzw. die gesamte Schleife zu beenden

- einige interne Kommandos

. datei	Kommandos aus <i>datei</i> ausführen und dann zurückkehren; <i>datei</i> wird ggf. über PATH gesucht; die Kommandos von <i>datei</i> werden innerhalb der aktuellen <i>Shell</i> ausgeführt, d.h. sie "sehen" lokale Variablen der <i>Shell</i>
break [n]	<i>for</i> - oder <i>while</i> -Schleife verlassen; falls <i>n</i> spezifiziert ist, <i>n</i> Schleifen verlassen
continue [n]	Iteration einer <i>for</i> - oder <i>while</i> -Schleife verlassen; falls <i>n</i> spezifiziert ist, wird mit einer Iteration der <i>n</i> -ten Schleife fortgesetzt
exec cmd	anstelle der <i>Shell</i> wird das Kommando <i>cmd</i> ausgeführt; es wird kein neuer Prozeß erzeugt
exit [n]	beendet die <i>Shell</i> ; Rückgabewert: <i>n</i>
export [name]	exportiert lokale Variablen in die globale Umgebung; ohne Parameter werden alle exportierten Variablen ausgegeben

hash [-r] [cmd]	die <i>Shell</i> merkt sich das Verzeichnis des Kommandos, so daß beim nächsten Aufruf nicht mehr der Suchpfad durchlaufen werden muß; "-r" bewirkt, daß alle Einträge in der <i>Hash</i> -Tabelle gelöscht werden
read name ...	es wird eine Zeile von <i>stdin</i> eingelesen, wobei die einzelnen Wörter den einzelnen Variablen zugewiesen werden; falls es mehr Wörter als Variablen gibt, erhält die letzte Variable den Rest der Zeile
return [n]	Rückkehr aus einer Funktion mit dem Wert <i>n</i> bzw. dem Rückgabewert des letzten Kommandos
shift [n]	die Werte der Positionsparameter werden um eine bzw. <i>n</i> Positionen nach links verschoben
test	auswerten eines Ausdrucks
times	gibt die akkumulierten Benutzer- und Systemzeiten aus
trap [cmd] [n] ...	<i>cmd</i> wird ausgeführt, wenn die <i>Shell</i> das Signal <i>n</i> empfängt; falls <i>cmd</i> fehlt, werden die angegebenen Signale wieder auf den ursprünglichen Wert zurückgesetzt (Beispiel: trap "echo 'ctrl-c gedrueckt'" 2)
unset [name ...]	die Variable <i>name</i> wird gelöscht
wait [n]	die <i>Shell</i> wartet auf das Ende von Hintergrundprozeß <i>n</i> ; falls <i>n</i> fehlt, wird auf das Ende aller Sohnprozesse der <i>Shell</i> gewartet

1.6.2 *C-Shell*

- entwickelt von *Bill Joy*
- sie hat ihren Namen wegen der Ähnlichkeit der *Shell*-Programmierung mit C
- bietet wesentlich mehr Komfort und Funktionalität als die *Bourne-Shell*
- steht unter */usr/bin/csh* als dynamisch gebundene *Shell* zur Verfügung
- Hintergrundprozesse eines Benutzers bleiben auch beim *Logout* erhalten !
- *Shell*-Skripte für die *C-Shell* laufen nur unter der *C-Shell* ab
- *Shell*-Variablen setzen / löschen
 - set name = wert
(vor und nach dem Gleichheitszeichen dürfen Leerzeichen stehen)
 - **Beispiel:** set xyz = "Alles klar?"
echo \$xyz ⇒ Alles klar?
csh
echo \$xyz ⇒

⇒ Variable ist nur lokal (in der aktuellen *Shell*) bekannt

- mit *setenv* können globale Umgebungsvariablen definiert werden

```
setenv abc Alles
```

```
setenv xyz "klar?"
```

Welche Ausgabe liefert: (env; set) | egrep "abc|xyz"

```
csh
```

```
echo "$abc $xyz" ⇒ Alles klar?
```

```
set abc = Na; set xyz = klar! (diese Änderung ist nur lokal!)
```

```
echo "$abc $xyz" ⇒ Na klar!
```

Welche Ausgabe liefert: (env; set) | egrep "abc|xyz"

```
csh
```

```
echo "$abc $xyz" ⇒ Alles klar?
```

```
exit
```

```
setenv abc Na; setenv xyz klar! (diese Änderung ist global!)
```

```
echo "$abc $xyz" ⇒ Na klar!
```

Welche Ausgabe liefert: (env; set) | egrep "abc|xyz"

```
csh
```

```
echo "$abc $xyz" ⇒ Na klar!
```

```
exit; exit
```

```
echo "$abc $xyz" ⇒ Alles klar?
```

Welche Ausgabe liefert: (env; set) | egrep "abc|xyz"

⇒ *setenv* hat nur eine Wirkung für Sohnprozesse und nicht für den Vaterprozeß

- mit *unset* bzw. *unsetenv* kann eine lokale bzw. globale Variable gelöscht werden, z.B. *unset abc*, *unsetenv xyz*
- lokale Variablen überdecken globale Variablen gleichen Namens

- Klammerung von Kommandos
 - (Kommandoliste)
 - ⇒ Kommandoliste in einer Sub-Shell ausführen
 - in der C-Shell können keine Funktionen definiert werden
 - ◆ in eingeschränkter Form können einige Dinge über das Kommando *alias* erreicht werden
 - ◆ **Beispiel:** alias dir = 'ls -al'
- Eingabe in ein Shell-Skript

```
#!/bin/csh -f
echo -n "Eingabe: " # -n verhindert <newline>
set eingabe = $<
echo "eingegebene Zeichenfolge: $eingabe"
```

- bedingte Ausführung

```
if (Bedingung) Kommando oder

if (Bedingung_1) then
  cmd_liste_1
[else if (Bedingung_2) then
  ...]
[else
  cmd_liste_n]
endif
```

- in der zweiten Variante **müssen** *if* und *then* auf derselben Zeile stehen
- optional können mehrere *else if*-Anweisungen folgen
- optional kann genau eine *else*-Anweisung folgen
- folgende Operatoren sind möglich
 - ◆ die logischen, arithmetischen und Vergleichsoperatoren der Sprache C
 - ◆ Operationen mit Dateien
 - r wahr: Datei existiert und *lesen* erlaubt
 - x wahr: Datei existiert und *ausführen* erlaubt
 - e wahr: Datei existiert
 - d wahr: Datei existiert und ist Verzeichnis
 - z wahr: Datei existiert, Dateigröße ist gleich 0
 - ...
- Verzweigung

```
switch (wort)
  case muster1:
    cmd_liste_1
    [breaksw]
  case muster2:
    cmd_liste_2
    [breaksw]
  ...
  default:
    cmd_liste_n
endsw
```

- *wort* wird in der Reihenfolge der Anweisungen mit den Mustern verglichen
 - bei Übereinstimmung wird die entsprechende *cmd_liste* ausgeführt
 - jede Alternative kann mit "breaksw" abgeschlossen werden (in diesem Fall wird die "switch"-Anweisung verlassen)
 - falls "breaksw" fehlt, werden, wie in der Sprache C üblich, die Anweisungen der anderen Alternativen ebenfalls ausgeführt
 - "default" trifft auf jedes *wort* zu, falls *wort* mit keinem Muster übereinstimmt
 - eine Kommandoliste kann auch für mehrere Muster zutreffen
case muster1a: case muster1b: ... case muster1n: ...
 - Muster dürfen die Zeichen "?" (ein beliebiges Zeichen), "*" (beliebige Anzahl Zeichen) und "[...]" (Zeichengruppe) enthalten
- *foreach*-Schleife

```
foreach var (Werteliste)
  cmd_liste
end
```

- "var" durchläuft alle Werte
- analog zur Sprache C kann *continue* bzw. *break* benutzt werden, um die aktuelle Iteration bzw. die gesamte Schleife zu beenden
- die Werteliste kann mit Hilfe von "?", "*" und "[...]" gebildet werden

- *while*-Schleife

```
while (Bedingung)
    cmd_liste
end
```

analog zur Sprache C kann *continue* bzw. *break* benutzt werden, um die aktuelle Iteration bzw. die gesamte Schleife zu beenden

- einige interne Kommandos

<code>alias [name [cmd]]</code>	<i>name</i> wird als Kürzel für <i>cmd</i> definiert; falls <i>cmd</i> fehlt, wird die Definition von <i>name</i> ausgegeben; falls beide Parameter fehlen, werden alle <i>alias</i> -Definitionen ausgegeben
<code>exec cmd</code>	anstelle der <i>Shell</i> wird das Kommando <i>cmd</i> ausgeführt; es wird kein neuer Prozeß erzeugt
<code>exit [n]</code>	beendet die <i>Shell</i> ; Rückgabewert: <i>n</i>
<code>history [n]</code>	gibt ggf. die letzten <i>n</i> Kommandos aus
<code>kill [-sig] [pid] [%job]</code>	bricht den angegebenen Prozeß / Job mit der angegebenen Signalnummer bzw. SIGTERM ab; "kill -l" gibt die symbolischen Namen aller Signale aus
<code>login [benutzer]</code>	beendet die Sitzung des aktuellen Benutzers und führt ein <i>LOGIN</i> für den angegebenen Benutzer durch
<code>logout</code>	beendet eine <i>Login-Shell</i>
<code>nice ...</code>	ändert die Priorität des Prozesses

popd [+n]	das oberste Element einer Verzeichnisliste wird zum aktuellen Verzeichnis; die Liste wird verkleinert; das oberste Element hat die Nummer 0
pushd [+n dir]	das aktuelle Verzeichnis wird an die Verzeichnisliste angefügt und ein " <i>cd dir</i> " ausgeführt; ohne Parameter wird das aktuelle Verzeichnis mit dem obersten Verzeichnis der Liste getauscht; bei "+n" wird das <i>n</i> -te Element zum obersten Element der Liste
rehash	baut die interne <i>Hash</i> -Tabelle der <i>Shell</i> neu auf
set [var [=wert]]	ohne Parameter werden die Werte aller <i>Shell</i> -Variablen ausgegeben; ohne <i>wert</i> wird der Variablen <i>var</i> der Wert "null" zugewiesen; <i>wert</i> kann ein einzelnes Wort, eine Zeichenfolge oder eine in Klammern eingeschlossene Liste von Wörtern sein (set a=xyz; set b="x y z"; set c=(a b c))
setenv [VAR [wert]]	analog <i>set</i> für Umgebungsvariablen; für Umgebungsvariablen werden i.a. nur Großbuchstaben benutzt
shift [var]	die Werte von <i>argv</i> oder <i>var</i> werden um eine Position nach links verschoben
source datei	Kommandos aus <i>datei</i> ausführen und dann zurückkehren; <i>datei</i> wird ggf. über PATH gesucht; die Kommandos von <i>datei</i> werden innerhalb der aktuellen <i>Shell</i> ausgeführt, d.h. sie "sehen" lokale Variablen der <i>Shell</i> ; ein Fehler an beliebiger Stelle beendet alle geschachtelten <i>source</i> -Anweisungen

<code>time [cmd]</code>	gibt die akkumulierten Benutzer- und Systemzeiten des Kommandos <i>cmd</i> bzw. der <i>Shell</i> aus
<code>unalias cmd</code>	löscht ein Kommando-Kürzel (" <code>unalias *</code> " löscht alle Kommando-Kürzel)
<code>unset [name ...]</code>	die Variable <i>name</i> wird gelöscht (" <code>unset *</code> " löscht alle <i>Shell</i> -Variablen ⇒ sehr unangenehme Seiteneffekte!)
<code>unsetenv name</code>	analog <i>unset</i> für Umgebungsvariablen
<code>wait</code>	die <i>Shell</i> wartet auf das Ende aller Hintergrundprozesse

Aufgabe 1-5:

Erstellen Sie unter der *Bourne*- und der *C-Shell* folgende Skripte:

- Erstellen Sie eine Kopierfunktion, die eine Datei in eine andere Datei kopiert. Die beiden Dateinamen können auf der Kommandozeile übergeben werden. Falls ein Dateiname fehlt oder sogar beide Dateinamen fehlen, erfragt das Skript die fehlenden Informationen.
- Erstellen Sie ein Skript, das rekursiv alle Dateien `"*.c"` einer Verzeichnisstruktur in `"*.c.bak"` kopiert. Legen Sie eine geeignete Verzeichnisstruktur an.
- Erstellen Sie ein Skript, das alle in b) erstellten Kopien in `"*.c.tmp"` umbenennt.
- Erstellen Sie ein Skript, das alle Dateien `"*.c.tmp"` nach Rückfrage löscht (der Befehl `"rm -i"` ist nicht erlaubt).

1.7 Konfiguration heterogener UNIX-Umgebungen

- Konfiguration hängt von der Art des Rechnernetzes ab
 1. vernetzte Einzelsysteme ohne NFS und NIS / NIS+, d.h. die Benutzer haben auf jedem Rechner ein eigenes *HOME*-Verzeichnis
 - ⇒ Benutzerverwaltung auf jedem Rechner erforderlich
(entfällt, wenn NIS / NIS+ benutzt wird)
 - ⇒ Konfiguration der *HOME*-Verzeichnisse auf jedem Rechner notwendig
(inkl. *~/.rhosts*, falls Anmeldung ohne Passwort möglich sein soll)
 - ⇒ für *lokale* Ausgaben eines Programms, das auf einem anderen Rechner läuft, muß unter *X Window* die Datei *~/.Xauthority* auf dem entfernten Rechner geeignet ergänzt werden, wenn nicht allen Programmen des entfernten Rechners der Zugriff auf den lokalen Bildschirm erlaubt werden soll

"xauth list" zeigt den Inhalt der Datei *~/.Xauthority* an:

```
tyr:0 MIT-MAGIC-COOKIE-1 55556b6554...
tyr/unix:0 MIT-MAGIC-COOKIE-1 55556b6554...
localhost:0 MIT-MAGIC-COOKIE-1 6bab4fef76...
hawkins.isc.anglia.ac.uk:0 MIT-MAGIC-COOKIE-1 467a756...
guirre.csi.ull.es:0 MIT-MAGIC-COOKIE-1 65ef9...
```

entferntem Rechner Zugriff auf lokalen Bildschirm erlauben:

```
xauth extract - tyr:0 | rsh oskar xauth merge -
xauth extract - tyr:0 | rsh -l sgross hawkins.isc.anglia.ac.uk xauth merge -
xauth extract - tyr:0 | rsh -l siegmarg guirre.csi.ull.es xauth merge -
```

2. homogener Rechnerverbund mit NFS und NIS / NIS+, d.h. ein *HOME*-Verzeichnis für alle Rechner

⇒ Benutzerverwaltung ist nur auf dem NIS / NIS+-*Server* erforderlich

⇒ das *HOME*-Verzeichnis muß nur auf dem NFS-*Server* eingerichtet werden, der die *HOME*-Verzeichnisse bereitstellt

⇒ da alle Rechner dasselbe *HOME*-Verzeichnis sehen, kann die Ausgabe eines Programms problemlos auf den lokalen Bildschirm umgeleitet werden

Umleitung der Ausgabe:

```
setenv DISPLAY <lokaler Rechnername>:0.0
setenv DISPLAY tyr:0.0
setenv DISPLAY tyr.informatik.fh-fulda.de:0.0
```

3. heterogener Rechnerverbund mit NFS und NIS / NIS+, wobei für jeden Rechnertyp eine eigene NFS / NIS/NIS+-Domäne existiert, d.h. ein *HOME*-Verzeichnis für jeden Rechnertyp

⇒ Benutzerverwaltung ist auf dem NIS / NIS+-*Server* für den jeweiligen Rechnertyp erforderlich

⇒ *HOME*-Verzeichnisse müssen auf den NFS-*Servern* für die verschiedenen Rechnertypen eingerichtet werden

⇒ *~/Xauthority* muß ggf. wie unter Punkt 1 angepaßt werden

4. heterogener Rechnerverbund mit NFS und NIS / NIS+, wobei für **alle** Rechnertypen nur eine NFS / NIS/NIS+-Domäne existiert, d.h. es gibt nur ein *HOME*-Verzeichnis für alle Rechner

⇒ Benutzerverwaltung ist nur auf dem NIS / NIS+-*Server* erforderlich

⇒ das *HOME*-Verzeichnis muß nur auf dem NFS-*Server* eingerichtet werden, der die *HOME*-Verzeichnisse bereitstellt

⇒ da alle Rechner dasselbe *HOME*-Verzeichnis sehen, kann die Ausgabe eines Programms auf einem beliebigen Rechner problemlos auf den lokalen Bildschirm umgeleitet werden

⇒ es muß sichergestellt sein, daß Programme für verschiedene Rechnertypen in verschiedenen Verzeichnissen gespeichert werden, z.B. in

```
$HOME/${SYSTEM_ENV}/${MACHINE_ENV}/bin  
(SYSTEM_ENV: SunOS, Linux, ...; MACHINE_ENV: sparc, x86, ...)
```

⇒ es muß sichergestellt sein, daß der richtige Pfad für die Programme eines Rechners benutzt wird

⇒ es muß sichergestellt sein, daß Programme ggf. ihre rechner-spezifischen Initialisierungsdateien erhalten

Problem: Ein Benutzer meldet sich auf Rechner A (z.B. unter SunOS Sparc) an, so daß die Initialisierungsdateien korrekt für A eingerichtet werden. Er meldet sich danach mit *telnet* oder *rlogin* auf Rechner B (z.B. unter SunOS x86) an. Die Initialisierungsdateien werden nun für Rechner B bereitgestellt. Wenn er sich jetzt an Rechner B abmeldet und dann ein Programm mit Initialisierungsdateien auf Rechner A aufruft, benutzt das Programm *falsche* Initialisierungsdateien. Dieses Problem ist an der FH Fulda zur Zeit noch nicht gelöst.

Mögliche Lösung: Beim *LOGIN* werden die Initialisierungsdateien umbenannt in `<Datei>.<fortlaufende Nummer>`. Danach werden die rechner-spezifischen Initialisierungsdateien erzeugt. Beim *LOGOUT* wird die fortlaufende Nummer bei den Initialisierungsdateien mit der höchsten Nummer entfernt. Eine fortlaufende Nummer ist notwendig, da der Rechner mehrfach mit *telnet* oder *rlogin* gewechselt werden kann.

- die Umgebung für einen neuen Benutzer befindet sich im Verzeichnis `/opt/global/skel`
- einige Aspekte der Konfiguration sollen anhand eines *LOGIN* ohne XDM oder CDE verdeutlicht werden
- zuerst wird unter der *C-Shell* die Datei `~/.cshrc` ausgeführt

```
# Datei: .cshrc                               Autor: S. Gross
# Datum: 03.12.1999
#
# Durch Entfernen der Kommentierung ('#' am Zeilenanfang) koennen
# verschiedene zusaetzliche Eigenschaften ausgewaehlt werden.
#
#
# zusaetzliche Programmpakete auswaehlen.
#
# !!! Java Workshop benutzt sein eigenes JDK. Mit "java -version" kann
# !!! bestimmt werden, welche Version augenblicklich eingestellt ist.
# !!! JDK11x, JDK12, JWS10 und JWS20 schliessen sich gegenseitig aus,
# !!! d.h.es darf nur ein Paket aktiviert werden!
#
# !!! OPENINTERFACE, SMARTELEMENTS und ELEMENTS_ENVIRON schliessen sich
# !!! gegenseitig aus, d.h. es darf nur ein Paket aktiviert werden!
#
# Pakete, die auf allen Maschinen zur Verfuegung stehen

#set GCCDIR = gcc-2.8.1
#set GCCDIR = egcs-1.1.2           # Nachfolger von gcc-2.8.1
#set GCCDIR = gcc-2.95.2         # Kombination von gcc und egcs
#set LAM      = lam-6.3           # Message Passing Interface
...
#
# den folgenden Eintrag auf keinen Fall veraendern, da er die allgemeine
# Umgebung und die Umgebungen fuer die oben ausgewaehlten Programmpakete
# einstellt !!!
#

source /opt/global/cshrc
```

```
#####
##### hier beginnt der private Teil #####
#####
...

# ACHTUNG: Unter XDM muss der Window Manager bei SunOS in der Datei
# .xinitrc.SunOS.{sparc, x86} eingestellt werden!
if ($SYSTEM_ENV == SunOS) then
    # setenv OW_WINDOW_MANAGER olwm
    # setenv OW_WINDOW_MANAGER olvwm
    ...
endif
...

# ACHTUNG: Unter XDM muss der Window Manager bei Linux in der Datei
# .xsession.Linux.x86 eingestellt werden!
if ($SYSTEM_ENV == Linux) then
    ...
endif

#
# eigene Umgebung nur einmal an die Umgebungsvariablen anfüegen.
#

if (! $?LOCAL_ENV) then
    set path = ( $path ${DIRPREFIX_LOCAL}/bin . )
    if ($?MANPATH) then
        setenv MANPATH ${MANPATH}:${DIRPREFIX_LOCAL}/man
    else
        setenv MANPATH ${DIRPREFIX_LOCAL}/man
    endif
    ...
setenv LOCAL_ENV
endif

#
# aus "INCLUDE-" und "LIBRARY-PATH" Angaben fuer Compiler-Optionen -I und
# -L bilden ...
#

set I_DIRS = ( -I`echo $C_INCLUDE_PATH | sed 's:/:/ -I/g` )
setenv I_DIRS "$I_DIRS" # fuer Makefile
unset I_DIRS
...

setenv NNTPSERVER news.fh-fulda.de

set history = 60 # Anzahl Befehle, die sich die Shell merkt
set filec # Dateinamen vervollstaendigen (<Esc>, <Ctrl-d>)

#
# Einige Synonyme. Mit "\lp", "\gcc" usw. wird das Originalkommando ohne
# vordefinierte Kommandozeilenoption ausgefuehrt.
#

alias dir 'ls -al'
alias del 'rm -i'
alias copy 'cp -ip'
...
```

```

if ($SYSTEM_ENV == SunOS) then
  # Standarddrucker: Drucker haengt vom jeweiligen Labor ab
  alias lp      'lp -o nobanner -c'

  alias gcc     "gcc -ansi -pedantic -Wall -Wstrict-prototypes
                -Wmissing-prototypes -DSOLARIS -DSolaris -DSunOS $L_DIRS"
  alias cc     "cc -fast -fd -v -Xc -DSOLARIS -DSolaris -DSunOS
                $CC_I_DIRS $L_DIRS"
  alias gcc_mesa "gcc \!* -lglut -lMesaGL -lMesaGLU -lXmu -lXext -lXi
                -lX11 -lm"
  alias cc_mesa "cc \!* -lglut -lMesaGL -lMesaGLU -lXmu -lXext -lXi
                -lX11 -lm"
endif
...

#
# ggf. noch $HOME/.mycshrc ausfuehren
#

if (-e $HOME/.mycshrc) then
  source $HOME/.mycshrc
endif

```

- die globale Umgebung wird über */opt/global/cshrc* eingestellt

```

setenv SYSTEM_ENV `uname -s`
if ($SYSTEM_ENV == Linux) then
  setenv MACHINE_ENV `uname -m`
else
  setenv MACHINE_ENV `uname -p`
endif

if ( ($MACHINE_ENV == i386) || ($MACHINE_ENV == i486) || \
     ($MACHINE_ENV == i586) || ($MACHINE_ENV == i686) ) then
  setenv MACHINE_ENV "x86"
endif

setenv DIRPREFIX_GLOBAL    /opt/global
setenv DIRPREFIX_SHARE    /opt/prog/share
setenv DIRPREFIX_EXAMPLES /opt/prog/share/examples
setenv DIRPREFIX_PROG     /opt/prog/$SYSTEM_ENV/$MACHINE_ENV
setenv DIRPREFIX_LOCAL    $HOME/$SYSTEM_ENV/$MACHINE_ENV

# ggf. Datei mit Fehlermeldungen loeschen, damit keine veralteten
# Meldungen ausgegeben werden.
#
if (-e $HOME/.cshrc.error) then
  rm -f $HOME/.cshrc.error
endif

# globale Umgebung nur einmal an die Umgebungsvariablen anfragen

if (! $?GLOBAL_ENV) then
  # ++++++
  # ++++++ Betriebssystem- und architektur-abhaengige Umgebung ++++++
  # ++++++

  source $DIRPREFIX_GLOBAL/cshrc.$SYSTEM_ENV.$MACHINE_ENV

```

```

# ++++++
# ++++++ Umgebungen fuer waehlbare Programme festlegen ++++++
# ++++++

if ($?LAM) then                                # !!! alle Plattformen !!!
    source $DIRPREFIX_GLOBAL/lam.csh
endif
...
if ($?JDK11x) then
    if ( (! $?JDK12) && (! $?JWS10) && (! $?JWS20) ) then
        source $DIRPREFIX_GLOBAL/jdk1.1.x.csh
    else
        echo " " >> $HOME/.cshrc.error
        echo "\!\!\! Fehler in ~/.cshrc: Auswahl inkompatibler Pakete." \
            >> $HOME/.cshrc.error
        echo "\!\!\! Es wurde keine Umgebung fuer JDK11x eingestellt." \
            >> $HOME/.cshrc.error
        echo "\!\!\! Bitte korrigieren Sie die Datei ~/.cshrc \!" \
            >> $HOME/.cshrc.error
        echo " " >> $HOME/.cshrc.error
    endif
endif
...
setenv GLOBAL_ENV
endif

# ++++++
# ++++++ Sonstige Festlegungen ++++++
# ++++++

# Default Editor, ...
setenv EDITOR    xemacs
setenv CSHEDIT   emacs

setenv HOST_NAME `hostname | sed 's/\..*//'\`
if (! $?HOSTNAME) then
    setenv HOSTNAME $HOST_NAME
endif

alias setprompt 'set prompt="$HOST_NAME $cwd:t \!\! ";\\
                if (x$cwd == x"/")set prompt="$HOST_NAME / \!\! ";'

alias cd 'if ("!\:0-$" == cd) cd;\\
          if ("!\:0-$" != cd) cd \!$;\\
          setprompt'

setprompt

unset autologout >&! /dev/null

umask 022                                # nur "read, execute"-Rechte fuer "group, other"

if ( ($SYSTEM_ENV == SunOS) || ($SYSTEM_ENV == Linux) ) then
    alias netscape '/usr/local/lib/netscape/netscape'
    setenv MOZILLA_HOME ${DIRPREFIX_PROG}/netscape
endif
...

```

- die rechner-spezifische globale Umgebung wird über */opt/global/cshrc.\$SYSTEM_ENV.\$MACHINE_ENV* eingestellt

```
# Datei: cshrc.SunOS.sparc          Autor: S. Gross
# Datum: 08.09.1999
#

setenv OPENWINHOME /usr/openwin
setenv MOTIFHOME   /usr/dt
setenv PHIGSHOME  /opt/SUNWits/Graphics-sw/sunphigs-3.0
setenv XGLHOME    /opt/SUNWits/Graphics-sw/xgl-3.0
setenv XILHOME    /opt/SUNWits/Graphics-sw/xil
setenv TEXDIR     /opt/prog/teTeX
setenv TEXDIR_BIN $TEXDIR/bin/sparc-solaris2`uname -r |sed 's/\.*.//'\`"

# Verschiedene Suchpfade und Umgebungen festlegen.

# ++++++ PATH festlegen ++++++
if ($USER == root) then
  set path = ( /sbin /usr/sbin /usr/sadm/install/bin ... )
else
  set path = ( /usr/bin $OPENWINHOME/bin $MOTIFHOME/bin /usr/ccs/bin \
    /opt/SUNWspro/bin /opt/SUNWmfwm/bin /opt/SUNWguide/bin \
    /opt/SUNWgmfu/bin /opt/SUNWsunsol/bin /opt/SUNWrtvc/bin \
    /usr/ucb /usr/xpg4/bin /usr/proc/bin \
    $PHIGSHOME/on-line-tutorial $TEXDIR_BIN \
    ${DIRPREFIX_PROG}/bin ${DIRPREFIX_PROG}/pbmplus \
    /usr/local/bin )
endif

# ++++++ MANPATH festlegen ++++++
...

# ++++++ Umgebungen fuer allgemeine Programame festlegen ++++++
# ++++++
# ++++++

# ++++++ Umgebung fuer gcc festlegen ++++++
source ${DIRPREFIX_GLOBAL}/gcc_egcs.csh
...

# ++++++ Sonstige Festlegungen ++++++
# ++++++
# ++++++

# Default Pager, ...
setenv PAGER      more
setenv XKEYSYMDB  $OPENWINHOME/lib/X11/XKeysymDB

# Bei "Terminals" mit "Backspace"-Taste Zeichen loeschen. Bei "rsh" ist
# TERM nicht definiert, so dass keine Fehlermeldung erzeugt wird.
if ($?TERM) then
  stty erase `^H'      > /dev/null
endif

setenv CLASSPATH
```

- benutzer-spezifische Einstellungen

```
# Damit Aenderungen/Erweiterungen der Datei ~/.cshrc einfach uebernommen
# werden koennen, koennen eigene "alias"-Eintraege oder aehnliches in
# dieser Datei definiert werden. Auf diese Weise muessen bei neuen
# Versionen der Datei .cshrc nur die entsprechenden Kommentarzeichen
# entfernt werden, um ein spezielles Software-Paket oder einen speziellen
# Window-Manager auszuwaehlen.
#
# Achtung: Achten Sie darauf, dass die letzte Zeile mit einem
#           Zeilenvorschubendet, da die Anweisung andernfalls nicht
#           erkannt wird.
#
#
# Datei: .mycshrc                               Autor: S. Gross
# Datum: 28.05.1999
#

alias clean 'rm *.lj *.toc *.lof *.lot *.log *.dlg *.ilg *.idx *.aux *.ind'
```

- nach ~/.cshrc wird ~/.login ausgeführt

```
# Datei: .login                               Autor: S. Gross
# Datum: 02.10.1998
#
#
# Den folgenden Eintrag auf keinen Fall veraendern !!!
#

source $DIRPREFIX_GLOBAL/login

#
# Initialisierungsdatei und Menues fuer OpenWindows einstellen
#

if (-e $HOME/.openwin-init.$SYSTEM_ENV.$MACHINE_ENV) then
  cp -p $HOME/.openwin-init.$SYSTEM_ENV.$MACHINE_ENV $HOME/.openwin-init
else
  if (-e $HOME/.openwin-init) then
    rm -f $HOME/.openwin-init
  endif
endif

if (-e $HOME/.openwin-menu.$SYSTEM_ENV.$MACHINE_ENV) then
  cp -p $HOME/.openwin-menu.$SYSTEM_ENV.$MACHINE_ENV $HOME/.openwin-menu
else
  if (-e $HOME/.openwin-menu) then
    rm -f $HOME/.openwin-menu
  endif
endif
...

```

```
#
# MIME-Dateien fuer Netscape einstellen
#

if (-e $HOME/.mailcap.$SYSTEM_ENV.$MACHINE_ENV) then
  cp -p $HOME/.mailcap.$SYSTEM_ENV.$MACHINE_ENV $HOME/.mailcap
else
  if (-e $HOME/.mailcap) then
    rm -f $HOME/.mailcap
  endif
endif

if (-e $HOME/.mime.types.$SYSTEM_ENV.$MACHINE_ENV) then
  cp -p $HOME/.mime.types.$SYSTEM_ENV.$MACHINE_ENV $HOME/.mime.types
else
  if (-e $HOME/.mime.types) then
    rm -f $HOME/.mime.types
  endif
endif

#
# systemspezifische Einstellungen
#

if ($SYSTEM_ENV == SunOS) then
  setenv MBOX $HOME/Mail/mbox
  if (`tty` == "/dev/console") then
    if ("$TERM" == "sun" || "$TERM" == "AT386") then
      if (${?OPENWINHOME} == 0) then
        setenv OPENWINHOME /usr/openwin
      endif
      clear          # get rid of annoying cursor rectangle
      echo ""
      echo -n "Starting OpenWindows in 5 seconds (Ctrl-C to interrupt)"
      sleep 5
      $OPENWINHOME/bin/openwin
      clear          # get rid of annoying cursor rectangle
      echo -n "Automatically logging out in 5 s (Ctrl-C to interrupt)"
      sleep 5
      logout        # logout after leaving windows system
    endif
  endif
endif

if ($SYSTEM_ENV == IRIX) then
  if (! ${?ENVONLY}) then
    # Set the interrupt character to Ctrl-c and do clean backspacing.
    if (-t 0) then
      stty intr '^C' echoe
    endif
    # Set the TERM environment variable
    eval `tset -s -Q`
  endif
  # Set the default X server.
  if (${?DISPLAY} == 0) then
    if (${?REMOTEHOST}) then
      setenv DISPLAY ${REMOTEHOST}:0
    else
      setenv DISPLAY :0
    endif
  endif
endif
endif
```

- die Datei */opt/global/login*

```

# Bei SunOS und Linux erfolgt zunaechst ein LOGIN in UNIX und danach
# wird irgendwann ein Windowsystem gestartet (z.B. automatisch in der
# Datei "~/.login". Bei IRIX erfolgt das LOGIN dagegen ueber den "xdm",
# so dass dort die "News" nicht direkt unter UNIX ausgegeben werden
# koennen, sondern nur in einem Fenster. Dasselbe gilt fuer SunOS, wenn
# CDE benutzt wird.
#
# Datei: login                               Autor: S. Gross
# Datum: 08.06.1999
#
if ($SYSTEM_ENV == SunOS) then
# touch $DIRPREFIX_GLOBAL/news/Welcome
if (! -e $HOME/.last-news) then
  ls -rtCl $DIRPREFIX_GLOBAL/news/* | \
    awk -f $DIRPREFIX_GLOBAL/awk.cmd - > /tmp/${USER}-news
else
  find $DIRPREFIX_GLOBAL/news -type f -newer $HOME/.last-news | \
    awk -f $DIRPREFIX_GLOBAL/awk.cmd - > /tmp/${USER}-news
endif
set SIZE = `wc -c /tmp/${USER}-news | awk '{print $1}' -`
if ($SIZE > 33) then
  chmod 700 /tmp/${USER}-news
  if (! $?DT) then
    # login ueber Unix-Konsole (z.B. rlogin, telnet, kein CDE, ...)
    source /tmp/${USER}-news
  else
    # CDE Desktop-Login
    xterm -title "CAE-Labor" -geometry 100x44+400+200 \
      -e /tmp/${USER}-news
  endif
else
  rm -f /tmp/${USER}-news
endif
if (-e $HOME/.cshrc.error) then
  if (! $?DT) then
    # login ueber Unix-Konsole (z.B. rlogin, telnet, kein CDE, ...)
    more $HOME/.cshrc.error
  else
    # CDE Desktop-Login
    xterm -title "CAE-Labor" -geometry 100x44+400+200 \
      -e more -w $HOME/.cshrc.error
  endif
endif
endif
...

touch $HOME/.last-news
unset SIZE

```

- beim Verlassen der C-Shell wird `~/.logout` ausgeführt

```
clear

#
# Entfernen von unbeendeten Druckauftraegen
#

echo "Alle unbeendeten Druckauftraege werden geloescht."

if (($SYSTEM_ENV == SunOS) || ($SYSTEM_ENV == Linux)) then
  lprm `whoami`
endif

if ($SYSTEM_ENV == IRIX) then
  lpstat -o | grep `whoami` | awk '{print "cancel ", $1}' \
    > /tmp/${USER}-printjobs
  chmod 700 /tmp/${USER}-printjobs
  source /tmp/${USER}-printjobs
  \rm -f /tmp/${USER}*
  \rm -f /tmp/fsel.result          # LightWave Lockfile
endif
...
```

- falls die graphische Oberfläche über XDM oder CDE gestartet wird, ist der Ablauf etwas komplizierter

```
#!/bin/sh
#
# Auswahl einer lokalen Konfigurationsdatei fuer XDM (KDE).
#
# Unter XDM / KDE werden aus den globalen Konfigurationsdateien folgende
# lokale Konfigurationsdateien ausgefuehrt:
#
# IRIX:      /var/X11/xdm/Xsession          => ~/.xsession
# Linux:    /usr/X11R6/lib/X11/xdm/Xsession => ~/.xsession (xdm und kdm)
# SunOS:    /usr/openwin/lib/xdm/Xsession  => ~/.xinitrc
#
# Die UNIX-Konfigurationsdateien werden unter XDM / KDE auf den
# verschiedenen Betriebssystemen in unterschiedlicher Reihenfolge
# ausgefuehrt. Unter Linux verhalten sich XDM und KDE gleich.
#
# IRIX:      ~/.cshrc
#            ~/.login
#            ~/.xsession
#            ~/.logout wird nicht ausgefuehrt!
#
# Linux:    ~/.xsession
#            ~/.cshrc
#            ~/.login  wird nicht ausgefuehrt !
#            ~/.logout wird nicht ausgefuehrt!
#
```

```

# SunOS:      ~/.xinitrc
#             ~/.cshrc
#             ~/.login  wird nicht ausgefuehrt !
#             ~/.logout wird nicht ausgefuehrt!
#
#
# Unter SunOS sollte die Datei /usr/openwin/lib/xdm/Xsession geaendert
# werden, so dass sie ebenfalls die Datei ~/.xsession ausfuehrt.
#
# Unter IRIX kann der Window Manager in der Datei ~/.cshrc gewaehlt
# werden. Bei Linux und SunOS muss er in der Datei ~/.xsession.Linux.x86
# bzw. ~/.xinitrc.SunOS.{sparc, x86} oder ~/.xsession.SunOS.{sparc, x86}
# gewaehlt werden.
#
# Wenn die Datei ~/.login nicht ausgefuehrt wird, erhalten die Benutzer
# keine "Message of the day". Wenn die Datei ~/.logout nicht ausgefuehrt
# wird, werden die erforderlichen Aufraeumarbeiten (z.B. nicht
# ausgefuehrte Druckauftraege loeschen) nicht ausgefuehrt.
#
# Die Datei ~/.login bzw. /opt/global/login kann in der Datei
# ~/.xsession.Linux.x86 bzw. ~/.xinitrc.SunOS.{sparc, x86} oder
# ~/.xsession.SunOS.{sparc, x86} ausgefuehrt werden, bevor der Window
# Manager gestartet wird.
#
# Die Datei ~/.logout muss ueber die Datei /usr/X11R6/lib/X11/xdm/Xreset,
# /var/X11/xdm/Xreset oder /usr/openwin/lib/xdm/Reset (oder Xreset ?)
# gestartet werden. Da Xreset mit "Root"-Rechten arbeitet, ist das
# Ausfuehren von Benutzerdateien nicht unproblematisch, so dass die
# relevanten Teile aus ~/.logout direkt in Xreset bzw. Reset uebernommen
# werden sollten.
#
# Da XDM zur Zeit auf keinem Sun-Rechner eingesetzt wird, wird hierfuer
# augenblicklich auch keine Umgebung zur Verfuegung gestellt.
#
#
# Datei: .xsession                      Autor: S. Gross
# Datum: 08.10.1999
#

SYSTEM_ENV=`uname -s`
if [ $SYSTEM_ENV != "Linux" ]; then
    MACHINE_ENV=`uname -p`
else
    MACHINE_ENV=`uname -m`
fi

if [ $MACHINE_ENV = "i386" -o $MACHINE_ENV = "i486" -o \
    $MACHINE_ENV = "i586" -o $MACHINE_ENV = "i686" ]; then
    MACHINE_ENV="x86"
fi
if [ -x $HOME/.xsession.$SYSTEM_ENV.$MACHINE_ENV ]; then
    exec $HOME/.xsession.$SYSTEM_ENV.$MACHINE_ENV
fi

```

- Rechner-spezifische Konfigurationsdatei

```
#!/bin/sh
#
# lokale Konfigurationsdatei fuer XDM / KDE unter Linux. Unter KDE wird
# der Window-Manager im Anmelde-Menue von KDE ausgewaehlt, waehrend er
# unter XDM in dieser Datei festgelegt wird.
#
# Datei: .xsession.Linux.x86          Autor: S. Gross
# Datum: 08.10.1999

XDM=`cat /var/run/xdm.pid`
XDM=`ps --pid $XDM | egrep xdm`

if [ ! -z "$XDM" ]; then
#
# XDM laeuft
#
# Fuer "openwin" muss (!) einer der folgenden Window Manager ausgewaehlt
# werden (Standard: fvwm2 => kann von "openwin" nicht gestartet werden!
#                               => ein Login ueber XDM ist nicht moeglich!
#                               => Fehler ueber "rlogin" beheben!)
#
# WINDOWMANAGER=olwm              # nur fuer "openwin" aktivieren!
# WINDOWMANAGER=olvwm            # nur fuer "openwin" aktivieren!

# Ein Window Manager muss (!) gewaehlt werden.
#
# WIN_MANAGER=/usr/openwin/bin/openwin # WINDOWMANAGER (de)aktivieren!
# WIN_MANAGER=/usr/X11R6/bin/mwm
# WIN_MANAGER=/usr/X11R6/bin/twm      # startet mit leerer Oberflaeche
# WIN_MANAGER=/usr/X11R6/bin/ctwm    # startet mit leerer Oberflaeche
# WIN_MANAGER=/usr/X11R6/bin/fvwm2
if [ -z "$WIN_MANAGER" ]; then
    WIN_MANAGER=/usr/X11R6/bin/fvwm95
fi
export XDM=running
fi

SYSTEM_ENV=`uname -s`
if [ $SYSTEM_ENV != "Linux" ]; then
    MACHINE_ENV=`uname -p`
else
    MACHINE_ENV=`uname -m`
fi

if [ $MACHINE_ENV = "i386" -o $MACHINE_ENV = "i486" -o \
    $MACHINE_ENV = "i586" -o $MACHINE_ENV = "i686" ]; then
    MACHINE_ENV="x86"
fi

#
# Initialisierungsdatei und Menues fuer OpenWindows einstellen
#

if [ -e $HOME/.openwin-init.$SYSTEM_ENV.$MACHINE_ENV ]; then
    cp -p $HOME/.openwin-init.$SYSTEM_ENV.$MACHINE_ENV $HOME/.openwin-init
else
    if [ -e $HOME/.openwin-init ]; then
        rm -f $HOME/.openwin-init
    fi
fi
```

```

if [ -e $HOME/.openwin-menu.$SYSTEM_ENV.$MACHINE_ENV ]; then
  cp -p $HOME/.openwin-menu.$SYSTEM_ENV.$MACHINE_ENV $HOME/.openwin-menu
else
  if [ -e $HOME/.openwin-menu ]; then
    rm -f $HOME/.openwin-menu
  fi
fi
...

#
# Da unter Linux die Datei ~/.login von XDM nicht ausgefuehrt wird,
# muessen noch die "News" ausgegeben werden
#

xterm -geometry 100x44+200+10 -e /bin/csh /opt/global/login

if [ -z "$WIN_MANAGER" ]; then
  exec $WINDOWMANAGER
else
  exec $WIN_MANAGER
fi

```

- beim *LOGOUT* wird die Datei *Xreset* ausgeführt

```

#!/bin/sh
#
# Diese Datei wird als "Root" (!!!) ausgefuehrt, nachdem eine
# XDM / KDE-Sitzung beendet ist und bevor das "Display" geschlossen
# wird.
#
# Diese Datei muss auf allen (!) Linux-Rechnern im Verzeichnis
# "/usr/X11R6/lib/X11/xdm" unter dem Namen "Xreset" gespeichert
# werden.
#
#
# Datei: Xreset.Linux          Autor: S. Gross
# Datum: 08.10.1999
#

#
# Entfernen von unbeendeten Druckauftraegen
#

lprm `whoami`

#
# Sonstige Aufraeumarbeiten
#

#find /home/$USER -name core -type f -exec \rm -f {} \;

```

Aufgabe 1-6:

Erstellen Sie eine Umgebung für das *Common Desktop Environment* (CDE) von Sun Microsystems.

- a) Stellen Sie fest, welche Dateien in welcher Reihenfolge ausgeführt werden.
- b) Stellen Sie sicher, daß alle Aufgaben aus *~/.cshrc*, *~/.login* und *~/.logout* erledigt werden.
- c) Stellen Sie sicher, daß Ihre Skripte in einer heterogenen Umgebung mit gemeinsamer NFS-Domäne für die *HOME*-Verzeichnisse funktioniert.
- d) Stellen Sie sicher, daß alle Änderungen an der graphischen Oberfläche, die Sie auf einem Rechner vornehmen, auf **allen** Rechnern des Verbundes wirksam werden. Erstellen Sie eine Lösung analog zur Lösung für XDM unter IRIX.