

Partitionen und Dateisysteme

Lernziele

- Festplatten partitionieren können
- Linux-Dateisysteme auf einer Partition erstellen können
- Werkzeuge zur Dateisystemprüfung
- Lokale Dateisysteme in den Verzeichnisbaum einbinden
- Plattenplatz-Kontingentierung für Benutzer und Gruppen einrichten

Festplattenpartitionierung

- Festplatten lassen sich in kleinere Segmente aufteilen (Partitionen)
- Die Beschreibung dieser Aufteilung ist in der Partitionstabelle abgelegt
- Die erste Partitionstabelle ist im Sektor 0 der Platte zu finden
- Dieser Bereich heisst MBR (Master Boot Record)
- Es sind die ersten 512 Byte der Platte

Festplattenpartitionierung

- Die Partitionstabelle enthält vier Einträge
 - primäre Partitionen
- Wenn mehr Partitionen benötigt werden,
 - muss eine primäre Partition als erweiterte Partition angelegt werden
 - sie dient als Behälter für logische Partitionen
 - logische Partitionen stehen nicht in der Partitionstabelle (MBR)
 - sondern am Anfang der logischen Partition

Festplattenpartitionierung

- SCSI-Festplatten
 - Es können bis zu 15 Partitionen angelegt werden
- IDE-Festplatten
 - Es können bis zu 63 Partitionen angelegt werden
- SATA-, FireWire- und USB-Festplatten
 - werden von Linux wie SCSI-Festplatten behandelt

Festplattenpartitionierung

- Bevor man eine Platte mit *fdisk* partitioniert, muss man sich über die Aufteilung Gedanken machen
- Wieviel Platz sollte für Linux zur Verfügung gestellt werden?
- Was muss sonst noch auf die Platte?
 - Home-Verzeichnis
 - Auslagerungsdatei
 - Datenbank oder ein anderes Betriebssystem
- Wie sollte dieser Platz aufgeteilt werden?

Übungen

- [4.1] Was verstehen Sie unter dem Begriff »Partition«?
- Welche verschiedenen Partitionstypen gibt es?
- [4.2] Welche Anzahl der einzelnen Partitionstypen lassen sich auf einer Festplatte maximal unterbringen?

Anlegen einer neuen Partition

- Es muss noch freier Platz auf der Festplatte vorhanden sein
- Eine neue Partition kann im laufenden Betrieb angelegt werden
- Bestehende Partitionen können nicht ohne weiteres geändert werden
- Es können auch Partitionen für fremde Betriebssysteme angelegt werden
 - z.B. für Multi-Boot Systeme

Anlegen einer neuen Partition

- 1 Sichern der aktuellen Bootsektoren und Daten auf der betroffenen Festplatte
- 2 Aufteilen des Plattenplatzes mit *fdisk*
- 3 Anlegen des Dateisystems (Formatieren)
- 4 Einbinden der Partition mit *mount* bzw. */etc/fstab*

Anlegen einer neuen Partition

- Die Daten und Bootsektoren können unter anderem mit dem Kommando *dd* gesichert werden
- z.B. die gesamte Festplatte *hda* auf Tape
 - `dd if=/dev/hda of=/dev/st0a`
- oder den MBR in eine Datei
 - `dd if=/dev/hda of=mbr_hda.dat \`
`bs=512 count=1`

Anlegen einer neuen Partition

- Die Laufwerke werden über die entsprechenden Gerätedateien angesprochen
- etwa */dev/hda* für die erste IDE-Platte
 - hda = Primary IDE Master
 - hdb = Primary IDE Slave
 - hdc = Secondary IDE Master
 - hdd = Secondary IDE Slave
- oder */dev/sda* für die erste SCSI-Platte

Anlegen einer neuen Partition

- Der Name einer Partition entspricht dem Gerätenamen der Festplatte mit nachgestellter Partitionsnummer
- etwa `/dev/hda1` für die erste Partition auf der ersten IDE-Platte
- Primäre und die erweiterte Partitionen haben die Nummern 1 bis 4
- Die erste logische Partition hat immer die Nummer 5, egal wie viele primäre Partitionen auf der Platte sind

Anlegen einer neuen Partition

- *fdisk* ist ein interaktives Programm zur Manipulation der Partitionstabellen von Festplatten
- Anzeigen der Partitionstabelle
 - `fdisk -l /dev/hda`
- Die Partitionsgrößen in Sektoren anzeigen
 - `fdisk -lu /dev/hda`
- interaktiver Mode
 - `fdisk /dev/hda`

Anlegen einer neuen Partition

- Die Bedienung des Programms *fdisk* ist selbsterklärend
- Die wichtigsten Kommandos sind:
 - m zeigt die Kommandoübersicht
 - d löscht eine ausgewählte Partition
 - n legt eine neue Partition an
 - p zeigt die aktuelle Partitionstabelle an
 - t ändert die “partition system id“
 - q beendet das Programm, ohne Änderung
 - w schreibt die neue Partitionstabelle auf die Festplatte und beendet das Programm

Anlegen einer neuen Partition

- Eine neue Partition könnte dann folgendermassen angelegt werden:

```
# fdisk /dev/hda
Command (m for help): n
e extended
p primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-1869, default 1):
Using default value 1
Last cylinder or +size (1-1255, default 1255): +1G
Using value 255
```

Anlegen einer neuen Partition

- Eine aktuelle Partitionstabelle könnte dann folgendermassen aussehen:

```
# fdisk /dev/hda
Command (m for help): p
Disk /dev/hda: 255 heads, 63 sectors, 1869 cylinders
Units = cylinders of 16065 * 512 bytes

Device      Boot Start   End  Blocks  Id System
/dev/hda1   *         1    255  2048256  6 FAT16
/dev/hda2             256    260   40162+  83 Linux
/dev/hda3             261    277  136552+  82 Linux swap
/dev/hda4             278   1869 12787740  83 Linux
```

Anlegen einer neuen Partition

- Neben *fdisk* gibt es bei Linux noch andere Möglichkeiten, die Partitionierung zu ändern

- *cdisk* bildschirmorientiert

- *sdisk* nicht interaktiv, scriptfähig

- Es gibt auch grafische Tools

- *parted* GNU

- *YaST2* SuSE

- *DiskDruid* RedHat

Übungen

- [4.3] In Ihrem System befinden sich zwei Festplatten, die als Master und Slave an den IDE-Controller 0 angeschlossen sind. Die Master-Platte hat 120 GByte Kapazität und besitzt zwei primäre und zwei logische Partitionen von gleicher Größe, die 40-GByte-Slave-Platte ist in eine primäre und drei logische Partitionen gleicher Kapazität aufgeteilt.
- ? Welche Bezeichnungen haben diese Partitionen unter Linux?

Übungen

- [4.4] Lassen Sie sich mit fdisk die Partitionierungstabelle Ihrer Festplatte anzeigen.
- Legen Sie nach Anweisung Ihres Trainers eine neue ext2fs-Partition an.

Erstellen eines Dateisystems

- Eine neu angelegte Partition muss formatiert werden
- Es wird dabei eine Grundstruktur für das Dateisystem angelegt
- Unter Linux existiert dafür der Befehl
 - `mkfs`
- Der Dateisystemtyp wird mit der Kommandozeilenoption `-t` gewählt
 - `mkfs -t ext2 ...`

Anlegen von ext-Dateisystemen

- Z.B. ein Dateisystem mit 2048-Byte-Blöcken auf der zweiten logischen Partition der ersten IDE-Festplatte
- `mke2fs -b 2048 /dev/hda6`
 - `-b <Grösse>`
 - ein vielfaches von 512
 - typische Werte sind 1024, 2048, 4096
 - abhängig von der Partitionsgrösse

Anlegen von ext-Dateisystemen

● Weitere Optionen von *mke2fs*

- -c überprüft die Partition auf defekte Blöcke
- -i <Anzahl> setzt die »Inodedichte« fest
- -m <Anteil> Datenblöcke, die für root reserviert sind
- -j erzeugt zusätzlich ein Journal
- und damit ein ext3-Dateisystem!

Anlegen von ext-Dateisystemen

- Die Inode-Dichte legt fest, wie viele Dateien in einem Dateisystem angelegt werden können
- Jede Datei benötigt einen Inode
- Auch Gerätedateien, FIFOs oder kurze symbolische Links benötigen einen Inode
- ! Die Anzahl Inodes kann nicht mehr geändert werden

Anlegen von ext-Dateisystemen

- Mit der *mke2fs*-Option `-F` können auch Dateisystemobjekte erzeugt werden, die keine Blockgerätedateien sind
- Z.B. eine virtuelle CD-ROM
 - `# dd if=/dev/zero of=cdrom.img bs=1M count=650`
 - `# mke2fs -F cdrom.img`
 - `# mount -o loop cdrom.img /mnt`
- Die resultierende CD-ROM enthält ein echtes ext2-Dateisystem mit allen Rechten, Attributen, Zugriffskontrolllisten (ACLs) usw.

Anlegen von ext-Dateisystemen

- Bei einer Partition mit ext-Dateisystem können nachträglich noch Parameter geändert werden
 - `tune2fs [<Optionen>] <Gerät>`
- ! Dieses Kommando ist mit Vorsicht zu verwenden
- ! Das Dateisystem sollte keinesfalls zum Schreiben eingehängt sein

Anlegen von ext-Dateisystemen

Ein paar interessante Optionen

`-c` setzt Anz. Mounts zwischen Checks

`-e` Verhalten bei Fehlern

`continue` Normal weitermachen

`remount-ro` Schreiben verbieten

`panic` Kernel-Panik erzwingen

`-i` setzt Zeit zwischen Checks

`-l` zeigt Informationen im Superblock an

`-L` setzt einen Partitionsnamen

Anlegen eines Reiser-Dateisystemes

- Das Reiser-Dateisystem ist ein Journaling Dateisystem und kann mit dem Befehl
 - `mkreiserfs <Device>`
angelegt werden
- Änderung der logischen Blockgrösse von 4 kB sind erst ab Reiserfs 4 möglich
- Reiserfs 4 bis jetzt nicht im Kernel

Anlegen eines Reiser-Dateisystemes

- Informationen über Reiser-Dateisysteme auslesen

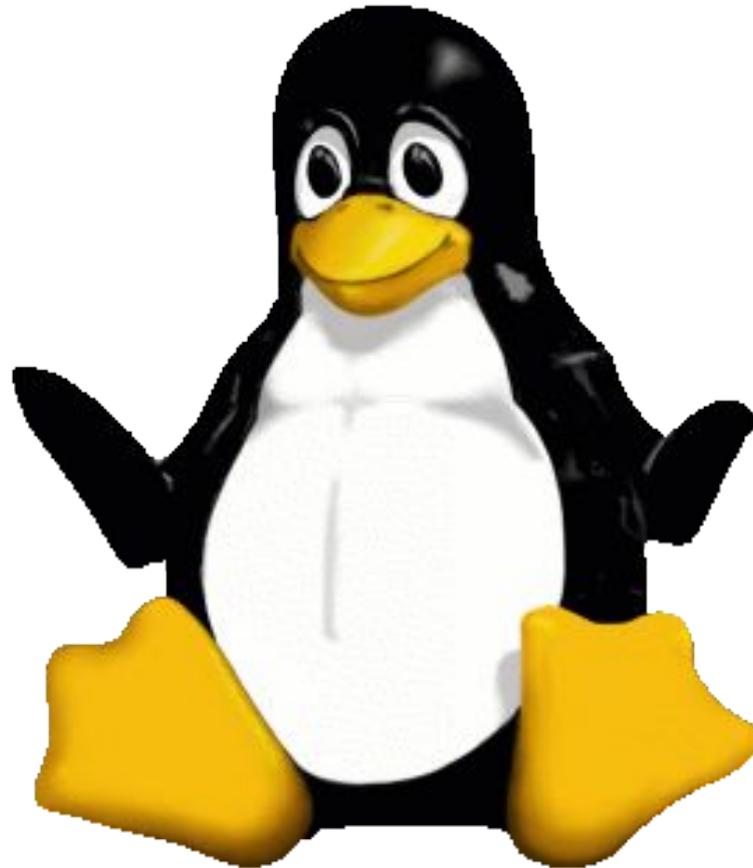
- `dumpreiserfs <Device>`

- Die Grösse einer Reiserfs-Partition kann geändert werden

- `resize_reiserfs [+|-] <size> <Device>`



Fragen?



Übungen

- ? [4.5] Ändern Sie die maximale Einhängenzahl der in Übung 4.4 angelegten Partition auf 30.
- ? Ausserdem sollen 30% des Speicherplatzes für den Benutzer *test* reserviert sein.

Übungen

-  [4.6] Wie lautet das Kommando, um die erste logische Partition der zweiten SCSI-Platte im Reiser-Dateisystem zu formatieren?

Reparatur von Dateisystemen

- Das Dateisystem kann sich in einen inkonsistenten Zustand befinden
 - Wenn der Rechner unkontrolliert ausgeschaltet wurde
 - Power Off
 - Stromausfall
 - oder wenn es zu einem Systemabsturz gekommen ist
 - Fehler in der Struktur des Dateisystems

Reparatur von Dateisystemen

- Fehler im Dateisystem können durch geeignete Programme erkannt und repariert werden
- Fehlerhafte Einträge in Verzeichnissen
- Fehlerhafte Einträge in Inodes
- Dateien, die in keinem Verzeichnis eingetragen sind
- Datenblöcke, die zu mehreren verschiedenen Dateien gehören

Reparatur von Dateisystemen

- Nicht alle Reparaturen können ganz ohne Datenverlust vorgenommen werden
- Das Dateisystem kann aber in der Regel wieder in einen konsistenten Zustand gebracht werden
- “verlorene“ Dateien werden in das Verzeichnis *lost+found* mit der Nummer der Inode eingetragen

Reparatur von Dateisystemen

- Jedes unter Linux verwendete Dateisystem benötigt sein spezielles Programm für den Dateisystemcheck
- Zu diesem Zweck existieren Systemprogramme wie
 - `fsck.vfat` oder `dosfsck`
 - `fsck.ext2` oder `e2fsck`
 - `fsck.reiserfs` oder `reiserfsck`
 - u. a.

Reparatur von Dateisystemen

- Die Prüfung und Reparatur aller Linux-Dateisysteme steuert
 - `fsck [<Optionen>] [-t <Typ>] <Device>`
- *fsck* dient zum Prüfen und Reparieren der Dateisystem-Struktur
- Für die Reparatur eines Dateisystems muss es ausgehängt oder Read-Only gemountet sein

Reparatur von Dateisystemen

● Zum Prüfen der zweiten Partition der zweiten IDE-Platte etwa

● `e2fsck -f /dev/hdb2`

● `-b <nr>` liest den Superblock mit `<nr>`

● `-f` erzwingt eine Prüfung

● `-c` sucht nach defekten Blöcken

● `-p` automatische Reparatur

● `-v` Informationen über den Status während dem Check

Reparatur von Dateisystemen

- Die einzelnen Schritte, die von e2fsck durchgeführt werden, sind:
 - 1 Die angegebenen Befehlsargumente werden geprüft.
 - 2 Es wird kontrolliert, ob das gewählte Dateisystem eingehängt ist.
 - 3 Das Dateisystem wird geöffnet.
 - 4 Es wird geprüft, ob der Superblock lesbar ist.
 - 5 Es wird geprüft, ob die Datenblöcke lesbar oder fehlerhaft sind.

Reparatur von Dateisystemen

(Fortsetzung)

- 6 Die Informationen aus dem Superblock über Inodes, Blöcke und Grössen werden mit dem aktuellen Zustand des Systems verglichen.
- 7 Es wird überprüft, ob die Verzeichniseinträge mit den Inodes übereinstimmen.
- 8 Es wird geprüft, ob jeder als belegt gekennzeichnete Datenblock existiert und genau einmal von einem Inode referenziert wird.
- 9 Die Anzahl der Links in den Verzeichnissen wird mit den Link-Zählern der Inodes verglichen (muss übereinstimmen).

Reparatur von Dateisystemen

(Fortsetzung)

- 10 Die Gesamtzahl der Blöcke muss der Anzahl der freien Blöcke plus der Anzahl der belegten Blöcke entsprechen.
- Dateien, deren Inodes in keinem Verzeichnis eingetragen sind, werden im Verzeichnis *lost+found* mit *Inodenummer* als Namen eingetragen
- Eine nicht leere Datei existiert, aber es gibt keinen Verweis aus irgend einem Verzeichnis. Inode-Nummer und Zugriffsrechte werden angezeigt und nachgefragt, ob die Datei gelöscht werden soll

Reparatur von Dateisystemen

- Nicht alle Fehler lassen sich einfach reparieren
- Was tun, wenn der Superblock nicht mehr lesbar ist?

● `# e2fsck -f -b 8193 /dev/hda2`

- oder mit entsprechenden Kenntnissen

● `# dumpe2fs`

● `# debugfs`

Reparatur von Dateisystemen

- Auch für das Reiser-Dateisystem existieren spezialisierte Programme zur Prüfung und Reparatur
 - # reiserfsck
 - # reiserfstune
 - # debugreiserfs
- Bei Reiserfs übernimmt der Kernel die Wiederherstellung anhand des Journals

Einbinden von Dateisystemen

- Dateisysteme dienen der Speicherung von Daten
- Prinzipiell kann man Dateisysteme direkt über die Gerätedateien ansprechen
- Dateioperationen (cp, mv und so weiter) können aber nur über den Verzeichnisbaum auf Dateien zugreifen
- Deshalb müssen Datenträger über ihre Gerätedateien in den Verzeichnisbaum eingehängt werden

Einbinden von Dateisystemen

- Das Einhängen wird mit dem Befehl *mount* vorgenommen
- Die Stelle im Verzeichnisbaum, an der ein Datenträger eingehängt wird, heisst Mountpunkt (engl. mount point)
- Es kann ein beliebiges Verzeichnis sein
- Das Verzeichnis muss nicht leer sein

Der mount-Befehl

- Einbinden von Dateisystemen

- `# mount [<Optionen>] <Gerät> <Mountpunkt>`

Beispiel:

- `# mount -t ext2 /dev/hda1 /mnt`

- Der einfache Befehl *mount* zeigt die momentan gemounteten Dateisysteme an

Der mount-Befehl

- In der Datei */etc/fstab* wird der Aufbau der gesamten Dateisystemstruktur aus verschiedenen Dateisystemen, die auf unterschiedlichen Partitionen, Platten usw. liegen können, beschrieben
- Hier werden auch Pseudodateisysteme wie *devpts*, *proc* und *usbfs* eingebunden
- Auch die Swap-Bereiche werden aktiviert
- */etc/fstab* wird beim Booten vom System abgearbeitet

Der mount-Befehl

Die Datei /etc/fstab (Beispiel)

/dev/hda3	/	ext3	defaults	1	1
/dev/hda5	/home	ext3	defaults	1	2
/dev/hda6	/data	reiserfs	defaults	0	2
/dev/fd0	/media/floppy	auto	noauto,user,sync	0	0
/dev/cdrom	/media/cdrom	auto	ro,noauto,user,exec	0	0
devpts	/dev/pts	devpts	defaults	0	0
proc	/proc	proc	defaults	0	0
/dev/hda1	swap	swap	pri=42	0	0

Der mount-Befehl

Die Datei /etc/fstab (Beispiel)

/dev/hda3	/	ext3	defaults	1	1
/dev/hda5	/home	ext3	defaults	1	2
/dev/hda6	/data	reiserfs	defaults	0	2
/dev/fd0	/media/floppy	auto	noauto,user,sync	0	0
/dev/cdrom	/media/cdrom	auto	ro,noauto,user,exec	0	0
devpts	/dev/pts	devpts	defaults	0	0
proc	/proc	proc	defaults	0	0
/dev/hda1	swap	swap	pri=42	0	0

Gerät

Der mount-Befehl

Die Datei /etc/fstab (Beispiel)

/dev/hda3	/	ext3	defaults	1	1
/dev/hda5	/home	ext3	defaults	1	2
/dev/hda6	/data	reiserfs	defaults	0	2
/dev/fd0	/media/floppy	auto	noauto,user,sync	0	0
/dev/cdrom	/media/cdrom	auto	ro,noauto,user,exec	0	0
devpts	/dev/pts	devpts	defaults	0	0
proc	/proc	proc	defaults	0	0
/dev/hda1	swap	swap	pri=42	0	0

Mountpoint

Der mount-Befehl

Die Datei /etc/fstab (Beispiel)

/dev/hda3	/	ext3	defaults	1	1
/dev/hda5	/home	ext3	defaults	1	2
/dev/hda6	/data	reiserfs	defaults	0	2
/dev/fd0	/media/floppy	auto	noauto,user,sync	0	0
/dev/cdrom	/media/cdrom	auto	ro,noauto,user,exec	0	0
devpts	/dev/pts	devpts	defaults	0	0
proc	/proc	proc	defaults	0	0
/dev/hda1	swap	swap	pri=42	0	0

Dateisystem

Der mount-Befehl

Die Datei /etc/fstab (Beispiel)

/dev/hda3	/	ext3	defaults	1	1
/dev/hda5	/home	ext3	defaults	1	2
/dev/hda6	/data	reiserfs	defaults	0	2
/dev/fd0	/media/floppy	auto	noauto,user,sync	0	0
/dev/cdrom	/media/cdrom	auto	ro,noauto,user,exec	0	0
devpts	/dev/pts	devpts	defaults	0	0
proc	/proc	proc	defaults	0	0
/dev/hda1	swap	swap	pri=42	0	0

Optionen

Der mount-Befehl

Die Datei /etc/fstab (Beispiel)

/dev/hda3	/	ext3	defaults	1	1
/dev/hda5	/home	ext3	defaults	1	2
/dev/hda6	/data	reiserfs	defaults	0	2
/dev/fd0	/media/floppy	auto	noauto,user,sync	0	0
/dev/cdrom	/media/cdrom	auto	ro,noauto,user,exec	0	0
devpts	/dev/pts	devpts	defaults	0	0
proc	/proc	proc	defaults	0	0
/dev/hda1	swap	swap	pri=42	0	0

dump

Der mount-Befehl

Die Datei /etc/fstab (Beispiel)

/dev/hda3	/	ext3	defaults	1	1
/dev/hda5	/home	ext3	defaults	1	2
/dev/hda6	/data	reiserfs	defaults	0	2
/dev/fd0	/media/floppy	auto	noauto,user,sync	0	0
/dev/cdrom	/media/cdrom	auto	ro,noauto,user,exec	0	0
devpts	/dev/pts	devpts	defaults	0	0
proc	/proc	proc	defaults	0	0
/dev/hda1	swap	swap	pri=42	0	0

fsck order

Der mount-Befehl

Optionen in */etc/fstab*

<code>/dev/fd0</code>	<code>/media/floppy</code>	<code>auto</code>	<code>noauto, user, sync</code>	<code>0</code>	<code>0</code>
<code>/dev/cdrom</code>	<code>/media/cdrom</code>	<code>auto</code>	<code>ro, noauto, user, exec</code>	<code>0</code>	<code>0</code>

- defaults Standardeinstellungen
- noauto das Dateisystem wird nicht eingebunden
- user normale Benutzer dürfen mounten
- sync direct write
- ro schreibgeschützt (read only)
- exec / noexec ausführbare Dateien können aufgerufen bzw. nicht ausgeführt werden

Fragen?



Übungen

- ❓ [4.7] Betrachten Sie die Einträge in den Dateien `/etc/fstab` und `/etc/mstab`.
- ❓ Wie unterscheiden sich diese?

Optimierte Partitionierung

- Platzbedarf
 - Die Grösse der gesamten Linux-Installation hängt von der geplanten Nutzung des Rechners ab
 - Neben dem Betriebssystem braucht es noch Platz für die benötigten Anwendungsprogramme
 - und für die zu verarbeitenden Daten

Optimierte Partitionierung

● Platzverteilung

- Die einfachste Lösung ist, nur eine Partition (plus Swap-Partition) für die gesamte Installation des Linux-Systems zu verwenden
- Sinnvoller ist jedoch eine Verteilung des Systems auf mehrere Partitionen
- Eine Minimalinstallation kann je nach Distribution weniger als 60MB belegen
- Eine aktuelle Distribution mit X, KDE und OpenOffice braucht leicht 2GB oder mehr

Optimierte Partitionierung

- Das hat folgende Vorteile:
 - Datensicherheit
 - Trennen von System- und Benutzerdaten
 - /usr und /opt auf einer extra Partition
»nur lesbar« mounten
 - Wartung
 - Einspielen von Updates
 - Benutzerdaten bleiben unverändert

Optimierte Partitionierung

- Weitere Vorteile:
 - Archivierung
 - Partitionen erleichtern das Anfertigen von Sicherheitskopien
 - Systemleistung
 - Durch Verteilung der Daten auf verschiedene Festplatten lassen sich wesentlich geringere Zugriffszeiten erzielen
 - “Überlaufschutz“
 - Verzeichnisse /tmp und /var

Optimierte Partitionierung

- Gerade für Server muss die Partitionierung dem Verwendungszweck entsprechend angelegt werden
- Zur Zeit der Partitionierung ist oft die aufkommende Datenmenge noch nicht bekannt
- Hierbei empfiehlt sich der Einsatz von LVM (Logical Volume Manager), um die Partitionierung flexibel halten zu können

Optimierte Partitionierung

- Unabhängig von den Partitionen sollte stets noch eine Swap-Partition erstellt werden
- In Zeiten kleiner RAM-Speicher wurde geraten, diese Partition mindestens doppelt so gross wie der vorhandene Arbeitsspeicher anzulegen
- Das würde heute zu Swap-Bereichen im GB-Bereich führen

Optimierte Partitionierung

- Grosse Swap-Partitionen machen höchstens bei Datenbanken noch Sinn
- Wenn Suspend-to-Disk vorgesehen ist, sollte der Swap etwas grösser als das RAM sein
- Bei Linux können bis zu 8 verteilte Swap-Partitionen eingebunden werden
- Notfalls kann auch zu Laufzeit eine Datei als Swap eingebunden werden

Optimierte Partitionierung

- Die Leistungsengestelle eines Linux-Systems ist sicherlich die Mechanik von Festplatten
- Durchschnittliche Zugriffszeiten auf eine Festplatte ist <10 ms
- Für den Arbeitsspeicher <10 ns
- ! Die Festplatte arbeitet also um den Faktor 1'000'000 langsamer
- So könnte für /tmp oder /var/spool z.B. eine Ramdisk verwendet werden

Optimierte Partitionierung

- Linux legt im Betrieb dynamische Puffer mit Festplattendaten im RAM an.
- Dort werden etwa mit *read ahead* bereits im Voraus gelesene Informationen abgelegt oder per *delayed write* Daten gesammelt, um diese später in einem Schritt auf den Datenträger zu schreiben
- Dies ist auch einer der Gründe, warum ein Linux-System nicht einfach während des Betriebs ausgeschaltet werden darf

Optimierte Partitionierung

- Die aktuelle Speicherbelegung kann mit dem Kommando *free* angezeigt werden

free

```
                total      used      free  shared  buffers  cached
Mem:            386620    380296    6324      0     6836   133280
-/+ buffers/cache: 240180 146440
Swap:          248968    169404    79564
```

Plattenkontingentierung (Quota)

- Die Maximalzahl von *Inodes* oder *Datenblöcken* kann pro Benutzer oder pro Gruppe beschränkt werden
- Es werden zwei Grenzen unterschieden
 - Die weiche Grenze (engl. soft quota)
 - Die harte Grenze (engl. hard quota)

Plattenkontingentierung (Quota)

- Kontingente können pro Dateisystem vergeben werden
- Z.B.
 - für die Grösse des Postfachs in /var/mail pro Benutzer
 - und eine andere Grösse für dessen Homeverzeichnis
 - und wieder eine andere Grösse für eine Gruppe in einem Projektverzeichnis

Plattenkontingentierung (Quota)

- Kontingente stehen für die gängigen Linux-Dateisysteme zur Verfügung
 - ext2
 - ext3
 - XFS
 - ReiserFS
- Die Programme zum Verwalten der Kontingente unterscheiden sich nach Dateisystem

Kontingentierung für Benutzer

- Um Kontingente für Benutzer einzurichten, muss eine entsprechende Software installiert sein (u.a. Kernelmodule)
- Die Dateisysteme müssen für die Kontingentierung gekennzeichnet sein

```
# mount -o remount,usrquota /home
```

oder in /etc/fstab

```
/dev/hda5 /home ext2 defaults,usrquota 0 2
```

Kontingentierung für Benutzer

- Die Kontingent-Datenbank muss mit
 - # quotacheck -avu
neu initialisiert werden, nachdem das Dateisystem mit der Option `usrquota` eingebunden wurde
- *quotacheck* legt im Wurzelverzeichnis des Dateisystems die Datenbank-Datei `aquota.user` an

Kontingentierung für Benutzer

● Das Kontingentierungssystem muss mit

● # quotaon -avu

gestartet werden

● # quotaoff -avu

stopt das Kontingentierungssystem wieder

● i.d.R wird das durch ein *Init-Script* erledigt

Kontingentierung für Benutzer

- *quotacheck* sollte auch im laufenden Betrieb periodisch ausgeführt werden, um die Datenbank zu »putzen«
- # *quotacheck*
- Z.B. einmal wöchentlich als Cronjob
- Nach jeder Änderung der Quota-Datenbank
- Bei jedem Systemstart

Kontingente verwalten

- Kontingente für verschiedene Benutzer setzt das Kommando

- `# edquota -u hugo`

- `# edquota -p tux hugo`

- Die Frist in Tagen, in der die Soft-Limit überschritten werden darf, wird pro Dateisystem definiert

- `# edquota -t`

Kontingente verwalten

- Benutzer können ihre Kontingente mit dem Kommando *quota* abfragen
 - # `quota`
 - # `quota -q`
- Die Option `-q` (quiet) gibt nur Dateisysteme aus, deren Grenze für Soft- oder Hardquota überschritten ist
- Eignet sich z.B. für `~/profile`

Kontingente verwalten

● *repquota* gibt einen tabellarischen Überblick über den Plattenplatzverbrauch verschiedener Benutzer

● # `repquota -a`

User	used	Block limits			File limits			
		soft	hard	grace	used	soft	hard	grace
root	-- 166512	0	0		19562	0	0	
tux	-- 2304	10000	12000		806	1000	2000	
hugo	-- 1192	5000	6000		689	500	1000	1.4w

Kontingentierungen für Gruppen

- Es können auch Gruppenkontingente vergeben werden
- Diese gelten dann für alle Mitglieder einer Gruppe
- Es müssen alle Gruppen, in denen Benutzer Mitglied sind, mit Kontingenten versehen werden
- Andernfalls kann man durch Gruppenwechsel die Quota umgehen

Kontingentierungen für Gruppen

- Für Gruppenkontingente wird die mount-Option *grpquota* verwendet

● `# mount -o remount,grpquota /project`

oder in `/etc/fstab`

```
/dev/hda5 /project ext2 defaults,grpquota 0 2
```

- Die Datenbank-Datei heisst hier `aquota.group`

Kontingentierungen für Gruppen

- Um die Gruppenkontingente verwenden zu können, müssen sie eingeschaltet werden mit

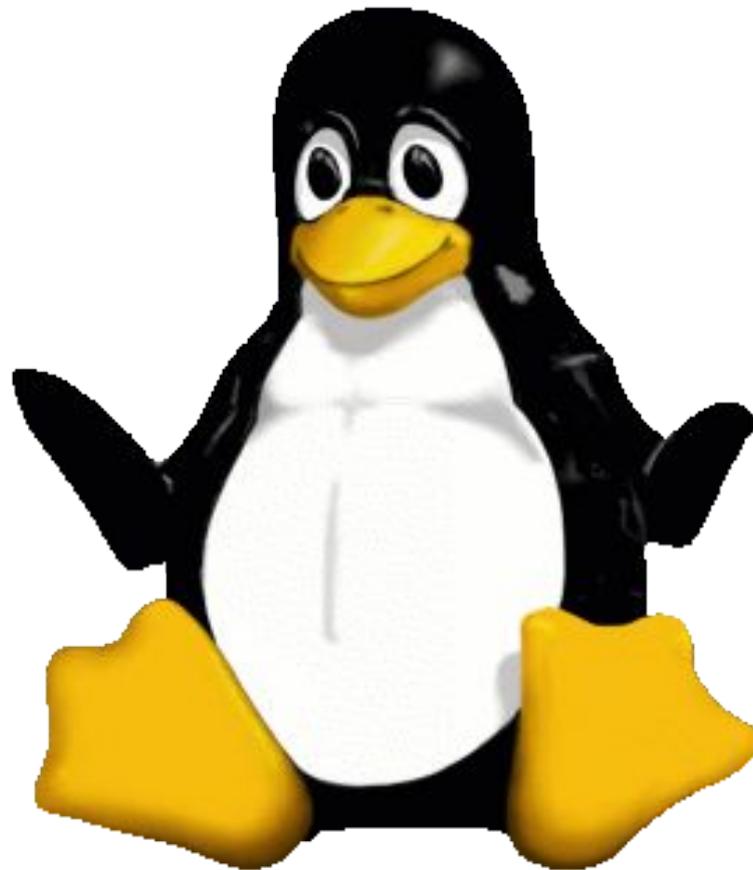
- # quotaon -augv

- Mit dem Kommando

- # quota -gv

werden die Gruppen-Kontingente angezeigt, in denen Sie Mitglied sind

Fragen?



Übungen

- [4.8] Richten Sie Plattenplatz-Kontingente wie eben beschrieben ein:
 - ein Dateisystem mit Benutzer-Quota,
 - ein Dateisystem mit Gruppen-Quota
 - und ein Dateisystem mit beiden.
- ! Achten Sie auf die Ausgabe von quotacheck, quotaon und quota.

Übungen

- [4.9] Legen Sie für einen Benutzer und für eine Gruppe (in der dieser Benutzer Mitglied ist) jeweils Beschränkungen an.
- ? Wie ändert sich die Ausgabe von quota, wenn Sie die weiche
- ? bzw. die harte Grenze erreichen.
- ? Bekommen Sie eine Warnung angezeigt?