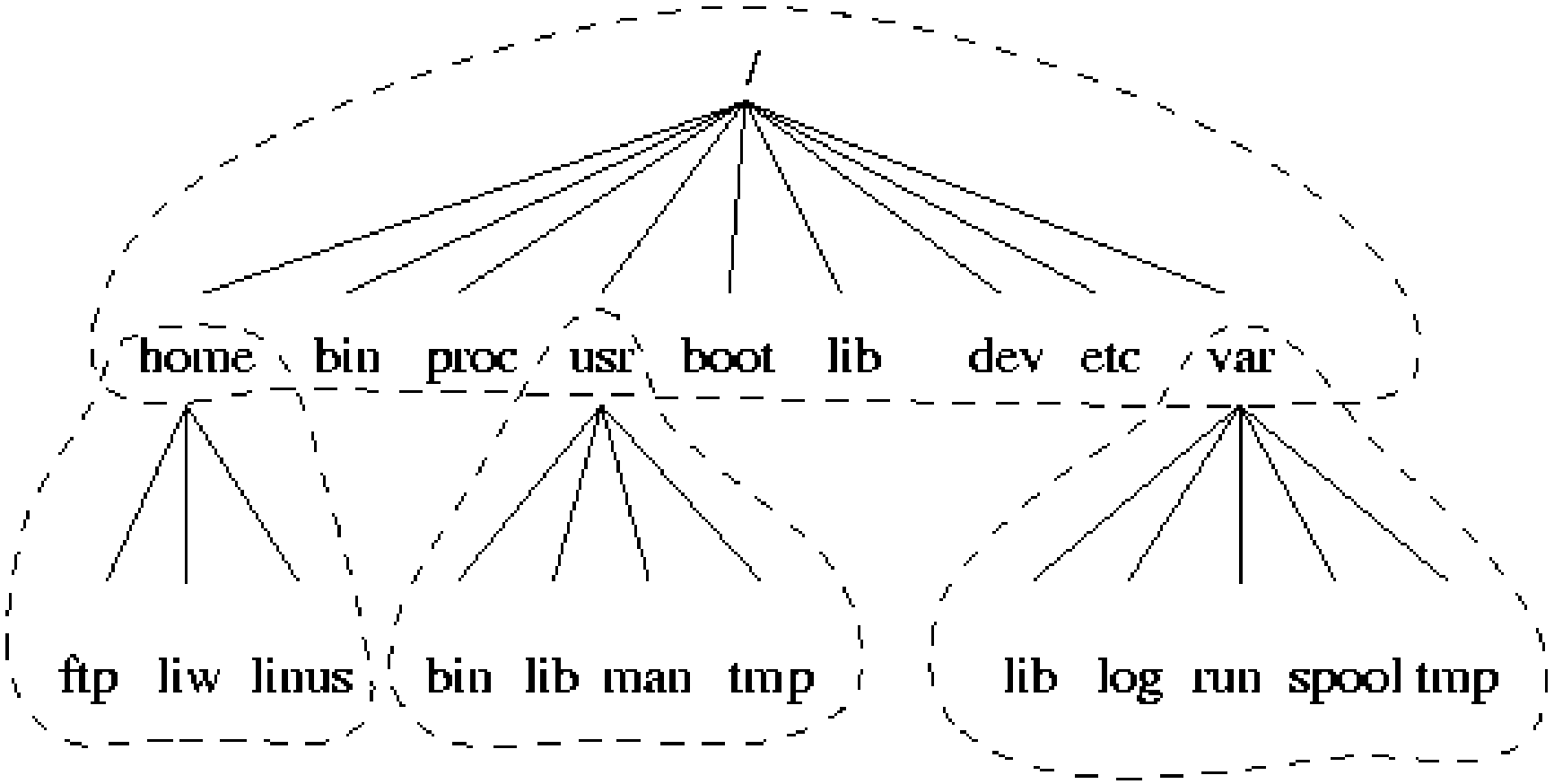
Failisüsteemide haldamine



Fail (*file*)

- Failid on organiseeritud ühte puusse, tipp on /
- Kõik on fail
  - fail, välisseade, mälu, protsessidevaheline kanal
- Faile on mitut tüüpi
  - tavaline fail, kataloog, erifail (seadmefail, toru, link, pistik)
- Igal failil on
  - andmeosa (võib ka puududa)
  - sõlm *(inode)*, milles on info faili kohta:
    - UID, GID, faili tüüp, viimase pöördumise, muutmise, sõlme muutmise, faili loomise aeg, faili suurus, juurdepääsuõigused, link andmeosale
  - vähemalt üks nimi mingis kataloogis

Kataloog

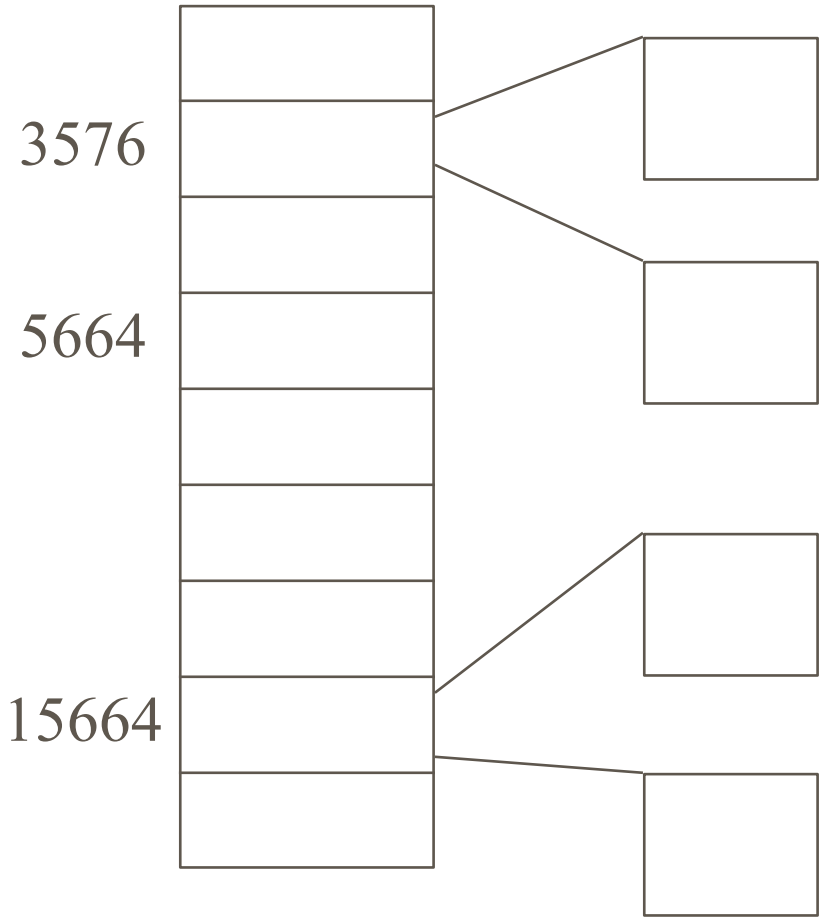
- Kataloog on eritüübiline fail
- Sisaldab failinime-sõlme paare
- Moodustavad hierarhilise struktuuri
- Igas kataloogis on kaks eritähendusega faili
  - . link kataloogi enda sõlmele
  - .. link vanem-kataloogi sõlmele

Kataloog, sõlm, andmed

Kataloog

fail1	5664
fail2	15664
fail3	3576
fail4	5664

Sõlmede tabel **Andmeplokid**



Seadmefailid

- Programmide jaoks on seadmed nagu tavalised failid
  - Nendesse saab kirjutada, neist saab lugeda
- Tavaliselt asuvad seadmefailid kataloogis /dev/
- Faili suuruse lahtris peetakse meeles seadmefaili 'major' ja 'minor' numbrid
- Major ja minor numbri kombinatsioon määrab kasutatava draiveri operatsioonisüsteemi tuumas
- Seadmefaile on kahte tüüpi:
  - sümbolseadmed – *character device* (terminal, printer), jadaseadmed
  - plokkseadmed – *block device* (kettad)

Jäik link *(hard link)*

- samale sõlmele võib viidata mitu kataloogikirjet
- sõlmedes peetakse arvet linkide arvu kohta
- fail kustutatakse füüsiliselt siis, kui
  - kustutatakse viimane link NING
  - fail on suletud
- saab teha ainult failidele samas failisüsteemis
- ei saa teha kataloogidele
- saab luua korraldusega

```
ln olemasolev_failinimi uus_failinimi
```

- võtab ruumi ainult kataloogikirjes
- Näiteks */bin/unzip* ja */bin/zipinfo, /bin/pigz* ja */bin/unpigz*

“Pehme” *(symbolic)* link

- Seob failinime olemasoleva failinimega
- On ise fail, mille sisuks on viidatav failinimi
- Võtab ruumi nii kataloogikirjes kui ka ≥1 andmeploki
- Algne (viidatav) fail “ei tea” pehme lingi olemasolust, faili ei pruugi isegi olemas olla
- Pehmel lingil puuduvad pääsuõigused
- Saab teha kataloogidele ja teistesse failisüsteemidesse
- Saab luua korraldusega

```
ln -s olemasolev_failinimi uus_failinimi
```

- Näiteks `/usr/tmp, /etc/init.d, /bin/sh`

Pääsukontroll

faili tüüp	omaniku õigused	grupi õigused	teiste õigused

r	lugemisõigus
w	kirjutamisõigus
x	käivitamisõigus
s	setUID, setGID
t	‘sticky’ bit

Loabittide tähendused

Loabitt	Tähendus failile	Tähendus kataloogile
r	Faili sisu saab lugeda	Saab kuvada kataloogi sisu (ls)
w	Faili sisu saab muuta (faili saab üle kirjutada)	Kataloogis saab faile luua, kustutada ning ümber nimetada
x	Faili (programm) saab käivitada	Kataloogi saab kasutada (sõlme leidmiseks) (cd)
SUID (s)	Programm käivitub faili omaniku õigustes	- (ei oma tähendust)
SGID (s)	Programm käivitub faili grupi õigustes	Kataloogi loodavad failid saavad kataloogi grupi
Sticky (t)	– (oli: programm jääb peale töö lõpetamist mällu)	Faile saab muuta ja kustutada vaid omanik

Pääsuõiguste muutmine (*chmod*)

- Õigusi saavad muuta ainult faili omanik ja root

```
chmod A#B failinimi
```

A — Kelle õigusi tahad muuta

u (omanik) g (grupp) o (teised) a (kõik)

\# — Kuidas tahad õigusi muuta

- (ära võtta) + (juurde anda) = (täpselt määratleda)

B — Õigus(ed)

r (lugemine) w (kirjutamine) x (käivitamine)
s (SUID, SGID) t (sticky bit)

Näiteks:

```
chmod ug+w failinimi
chmod o=rx failinimi
```

Sümbolilised ja numbrilised pääsuõigused

```
- r-- --- ---     0400
- -w- --- ---     0200
- --x --- ---     0100
- --- r-- ---     0040
- --- -w- ---     0020
- --- --x ---     0010
- --- --- r--     0004
- --- --- -w-     0002
- --- --- --x     0001
- --s --- ---     4000
- --- --s ---     2000
- --- --- --t     1000
```

Numbriline pääsuõiguste muutmine

	d	r	w	x	r	w	x	r	w	x	
		4	2	1	4	2	1	4	2	1	rwx bitid
				4			2			1	st bitid

```
chmod %### failinimi
```

\# on pääsuõigus numbrina (0-7)

```
  rwx = 7
  rw- = 6
  --- = 0
```

% on SUID, SGID, sticky pääsuõigus numbrina

```
  SUID+SGID = 6
```

Näiteks:

```
$ chmod 4754 filee
```

- faili „filee“ loabitid sätitakse

```
-rwsr-xr--
```

Uute failide pääsuõigused

`umask $$$` määrab pääsuõiguse uutele failidele

$$$ = 666 - pääsuõigus (failil)

$$$ = 777 - pääsuõigus (kataloogil)

Näiteks:

`$ umask 022` - uued failid tekivad õigustega 644

- uued kataloogid tekivad õigustega 755

`$ umask 002` - uued failid tekivad õigustega 664

- uued kataloogid tekivad õigustega 775

Omaniku muutmine (*chown*)

- Muudab faili omaniku ja/või grupi
- Faili omanik saab muuta ainult (faili) gruppi
  - omanik peab ise kuuluma uude gruppi
- Root saab muuta nii faili omanikku kui gruppi

```
chown  omanik[:grupp]  failinimi
chgrp  grupp  failinimi
```

Access Control Lists (ACL)

- Linux toetab ka ACL-e lisaks tavalisele loabittide-põhisele pääsukontrollile
  - Käsud

    `getfacl` - faili ACL vaatamine

    `setfacl` - faili ACL muutmine
  - Näide:

```
setfacl -m u:kasutaja:r failinimi
getfacl failinimi
```

  - Vaata ka:

```
man acl
man setfacl
```

ext[234] failisüsteemi atribuudid

`chattr mode fail(id)`

- Võimaldab anda failidele lisapiiranguid/omadusi
  - -a saab avada ainult lisamiseks (append)
  - -i read-only fail (immutable)
  - -S fail kirjutatakse kettale ilma puhverdamata (sync)

- On veel teisi atribuute ja neid lisandub tulevikus

  vt `man chattr`

- Lisaatribuutide vaatamine

`lsattr fail(id)`

Kasutatava kettaruumi loomine

- Kettaseadme ühendamine, jaotise loomine

  /dev/hd, /dev/sd*, /dev/md*, /dev/dm**

  fdisk /dev/sda - võimaldab redigeerida partitsioonitabelit

  parted /dev/sda - võimaldab redigeerida partitsioonitabelit

  lvm, system-config-lvm - LVM loogiliste kettaosade haldamine

- Failisüsteemi loomine

  mkfs -t vfat /dev/sdb

  mkfs -t etx4 /dev/sda3

- Failisüsteemi ühendamine

  mount /dev/sda4 /mnt/proov

Failisüsteemi loomine (*mkfs*)

```
mkfs -t type [options] device
```

- Linux toetab paljusid erinevaid failisüsteemi tüüpe
  - minix, ext, ext2, ext3, ext4, msdos, iso9660, vfat, ntfs, reiserfs, xfs, btrfs, ...
  - proc, devpts, tmpfs, ...
  - nfs, nfs4, smbfs, cifs, ncpfs,...
- `device` on seadmefail kataloogist */dev*

  /dev/sda1, /dev/fd0

Failisüsteemi ühendamine (*mount*)

- Failisüsteemi saab kasutada ainult siis kui ta on ühendatud (monteeritud, *mounted*)
- Failisüsteem (seadmefail) ühendatakse mingisse kataloogi, mida nim. *ühenduspunktiks* *(mount point)*.

`mount -t type -o op1,op2 seadmefail  ühenduspunkt`

- Võimalikud parameetrid (op1,op2)
  - *ro, rw, nodev* jt
- Lahtiühendamine

  `umount seadmefail` või `umount ühenduspunkt`
- Failisüsteemi kasutava protsessi väljaselgitamine, kui lahtiühendamine ei õnnestu

```
# fuser -mv /dev/seadmefail
```

Automaatne ühendamine (*/etc/fstab*)

```
/dev/sda3   /home   ext4   nodev,nosuid   1 2
```

- Määrab OS-i käivitumisel automaatselt ühendatavad failisüsteemid
- Vorming
  - failisüsteem (seadmefail)
  - ühenduspunkt
  - failisüsteemi tüüp
  - parameetrid
  - parameeter käsule *dump* (0, 1)
  - järjekorraparameeter failisüsteemi kontrollimise käsule *fsck*

Plokkseadmete nimed

- Plokkseadmete faile tekitab /dev-kataloogi alla udev-alamsüsteem. Dünaamiliselt.
- Klassikalised nimed
  - /dev/sda – esimene SCSI ketas
    - /dev/sda1, /dev/sda2 ... - partitsioonid seal
    - /dev/sdb – teine SCSI ketas
    - /dev/vda, /dev/vdb – virtuaaliseeritud ketas
- Probleem – ketta nimed ei pruugi jääda alati samaks
    - uue ketta ja/või -kontrolleri lisamisel
    - ketta tõstmisel teise kontrolleri või juhtme külge või ka nt SCSI-aadressi muutmisel

Plokkseadmete nimed

- Kas on ka midagi püsivat, mida kirjutada `/etc/fstab` faili?

- Variant1 – udev seadistustes sättida reeglitega kindlale seadmele kindel seadmefail (kasutatakse vähe)
  - udev seadistus asub kataloogis `/etc/udev` ja `/usr/lib/udev`
  - vt ka `man udev`

Plokkseadmete nimed

- Variant2 – udev-alamsüsteem tekitab dünaamiliselt veel 4 komplekti nimesid
  - LABEL – peaaegu igale failisüsteemile saab panna nime (salvestatakse failisüsteemi alguses) – udev loob seadmefailid kataloogi `/dev/disk/by-label`
  - ID – riistvara seerianumbri järgi – udev loob seadmefailid kataloogi `/dev/disk/by-id/`
  - PATH – riistvara kuuluvuse järgi alamsüsteemide alla – udev loob seadmefailid kataloogi `/dev/disk/by-path/`
  - UUID – failisüsteemile genereeritakse selle loomisel unikaalne nimi (Universally Unique IDentifier), mis peaks olema unikaalne vähemalt süsteemi piires ja püsiv (hoitakse failisüsteemi alguses). udev loob seadmefailid kataloog `/dev/disk/by-uuid/`
  - GPT partitsiooniskeemi kasutamine lisab veel kaks viitamise skeemi
    - by-partlabel
    - by-partuuid

Plokkseadmete nimed (*/etc/fstab* -s)

- Niisiis tuleks rea

```
/dev/sda3   /home   ext4   nodev,nosuid   1 2
```

- asemel kirjutada hoopis näiteks

```
UUID=650eb591-f4b6-41dc-918d-0f906ba34425   /home   ext4   nodev,nosuid   1 2
```

- Juhusliku UUID genereerib käsk
  - `uuidgen`
- Loodud failisüsteemi UUID saab vaadata olenevalt failisüsteemist
  - `tune2fs -l /dev/sda3` - ext2, ext3, ext4 korral
  - `xfs_admin -u /dev/sda3` - xfs korral
- Tuleb jälgida, et failisüsteemi muutmisel (nt suurendamisel) võib mõnel juhul UUID muutuda. Sel juhul tuleks `/etc/fstab` käsitsi muuta.

Kettakasutuse vaatamine

- Failisüsteemide kaupa

  `df [-i] [-h]`

  - i näita sõlmede kasutust
  - h näita arusaadavamas vormis (kilo-, mega-, gigabait)

- Kataloogide kaupa

  `du [-s] [-h] [...]`

  - s näita summaarset kettakasutust kataloogis
  - h näita arusaadavamas vormis (kilo-, mega-, gigabait)

Failisüsteemide parandamine

- tehakse **lahutatud(!)** failisüsteemiga
- *fsck* - kontrollib ja parandab failisüsteemi

```
fsck [-t type] [-a] failisüsteem
```

  - ext2,3,4 failisüsteemi korral pannakse leitud kodutud plokid failidesse kataloogis */lost+found*
- *badblocks* - kontrollib plokkseadme füüsilist olukorda
  - *Read-only* test – ainult loeb kettaplokke
  - *Read-write* test
    - *destructive* – algseid andmed ei säilitata!
    - *Non-destructive*

```
badblocks -v /dev/vg0/proov
```

Harjutus: kataloogi ümbertõstmine

Ül. Kasutajate kodukataloogide (/home) ümbertõstmine

uue jaotise tegemine	`fdisk /dev/sda`
failisüsteemi loomine	`mkfs -t ext4 /dev/sda3`
(failisüsteemi kontroll	`fsck /dev/sda3 )`
ajutine ühendamine	`mount /dev/sda3 /mnt`
failide kopeerimine	`cp -a /home/* /mnt`
identsuse kontroll	`diff -r /home /mnt`
/etc/fstab redigeerimine	

```
/dev/sda3   /home   ext4   defaults   1   2
```

vana ümbernimetamine	`mv /home /home.old ; mkdir /home`

ümbermonteerimine	`umount /mnt`
	`umount /home`
	`mount /home`

vana kustutamine	`rm -rf /home.old`

Harjutus: kataloogi ümbertõstmine

Ettevalmistus – LVM köitegrupi loomine

`fdisk /dev/sda` luua partitsioon /dev/sda4 üle kogu vaba ruumi (vajadusel restart)

```
# lvm
> pvcreate /dev/sda4          loome LVM füüsilise köite
> vgcreate vg0 /dev/sda4      loome LVM köitegrupi
> lvcreate -L 4G -n home2 /dev/vg0   loome loogilise köite 'home2'
> quit
#
```

Harjutus: kataloogi ümbertõstmine

Ül. /home kataloogipuu ümbertõstmine (LVM)

(uue köite tegemine `lvcreate või system-config-lvm`)

failisüsteemi loomine `mkfs -t xfs /dev/vg0/proov`

(failisüsteemi kontroll `xfs_repair /dev/vg0/proov`)

ajutine ühendamine `mount /dev/vg0/proov /mnt`

kodude kopeerimine `cp -a /home/* /mnt`

identsuse kontroll `diff -r /home /mnt`

/etc/fstab redigeerimine (et ühendataks automaatselt)

```
/dev/vg0/proov  /home  xfs  defaults  1  2
```

vana ümbernimetamine `mv /home /home.old ; mkdir /home`

(ümbermonteerimine `umount /mnt`
`umount /home`
`mount /home` avatud failid!)

Peale restarti peaks uus FS monteeritama /home alla `reboot`

Ketta kasutuskvoodid - *quota*

- Võimaldab piirata kettakasutust
  - Eraldi igas failisüsteemis
  - Kasutajate kaupa
  - Gruppide ühiste piirangute kaupa

- Mõisted

soft limit	piir, mille puhul hoiatatakse
hard limit	piir, millest üle minna ei saa
grace time	aeg mille jooksul võib “pehmet piiri” ületada

Kettakvootide kasutamine (ext[234] fs)

- failisüsteemi kvootide sisse/välja lülitamine
  - failisüsteem tuleb ühendada parameeteriga *usrquota* ja/või *grpquota*
    redigeerida faili */etc/fstab*
  - andmebaaside *aquota.user* loomine:
    ```
    # quotacheck -a
    ```
  - kvootide süsteemi reset:
    ```
    # quotaoff -a; quotaon -a
    ```
- piirangute ületamise kontrollimine:
  ```
  # repquota -a
  ```
- kettakvoodi seadmine
  ```
  # edquota kasutaja
  ```
  või
  ```
  # setquota kasutaja limiidid failisüsteem
  ```

Kettakvoodi seadmine XFS fs-ga

- Võimaldab sättida kvoote kasutaja, grupi ja/või projekti (kataloogihierarhia) põhiselt
- `man mount`
- `man xfs_quota`
- Abimaterjal

  https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/8/html/managing_file_systems/assembly_limiting-storage-space-usage-on-xfs-with-quotas_managing-file-systems#the-xfs_quota-tool_assembly_limiting-storage-space-usage-on-xfs-with-quotas

S.M.A.R.T.

Self-Monitoring, Analysis and Reporting Technology

Kõvaketta diagnostika ja seisukorra hindamine

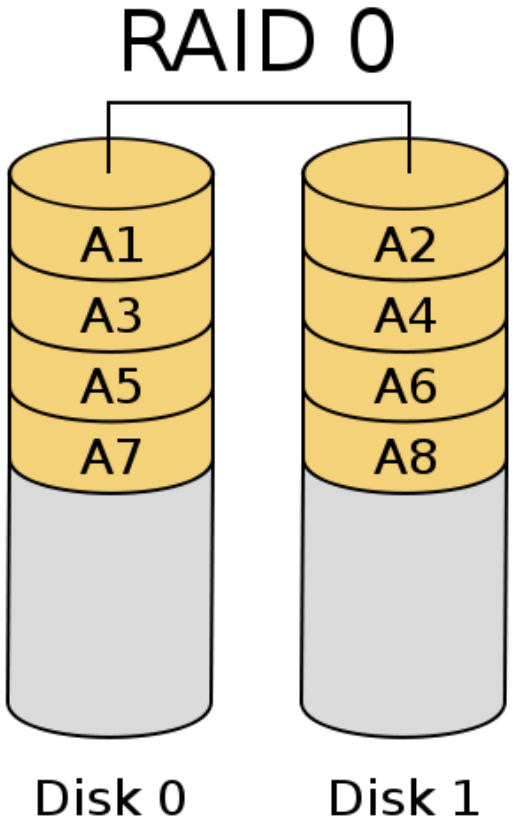
- CentOS/Rocky Linuxis pakk 'smartmontools'
- käsk `smartctl`
- `smartctl -a /dev/sda` – info
- `smartctl -t short /dev/sda` – ketta lühike test
- Vt `man smartctl`
- `smartd` – daemon SMART info monitoorimiseks, häiretest teavitatakse administraatorit e-maili teel

RAID

- Redundant Array of Inexpensive (Independent) Disks
- RAIDi tasemed
  - RAID 0
  - RAID 1
  - RAID 2 (ei kasutata)
  - RAID 3 (ei kasutata)
  - RAID 4 (ei kasutata või kasutatakse väga vähe)
  - RAID 5
  - RAID 6
  - RAID 0+1, RAID 10
  - RAID 15, RAID 51, RAID 60,...

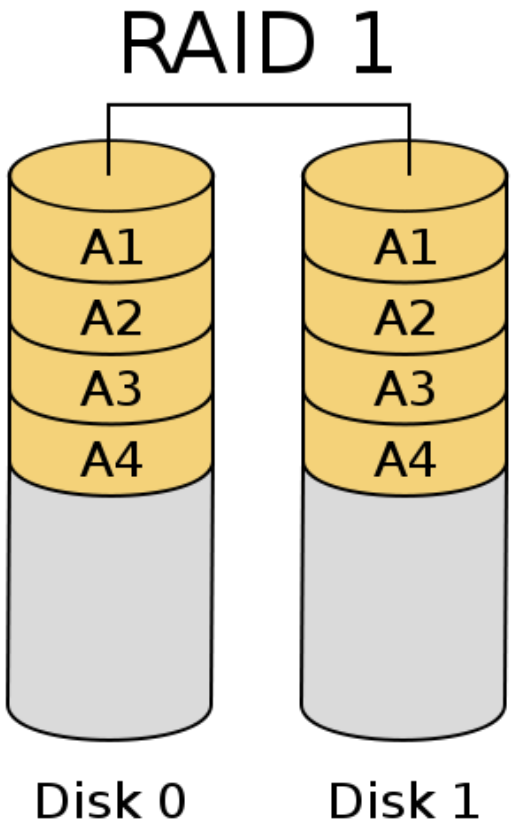
RAID 0

- Striping (block level)
  - Mahutavus: n
    - st n * väikseima ketta mahutavus
  - Kiirus
    - lugemisel – suurepärane
    - kirjutamisel – suurepärane
  - Veakindlus – olematu



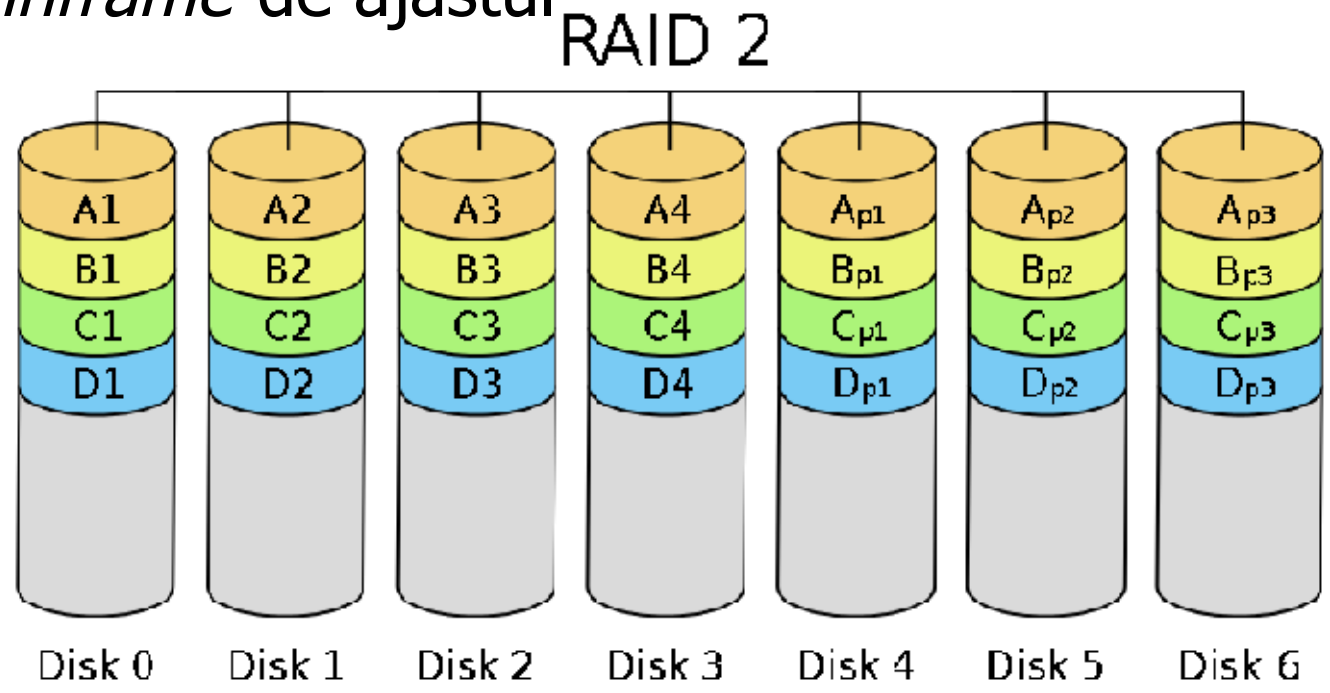
RAID 1

- Mirroring (Duplexing)
  - Mahutavus: väikseima ketta maht
  - Kiirus
    - lugemisel – suurepärane
    - kirjutamisel – väga hea
  - Veakindlus – andmed säilivad n-1 ketta riknemisel



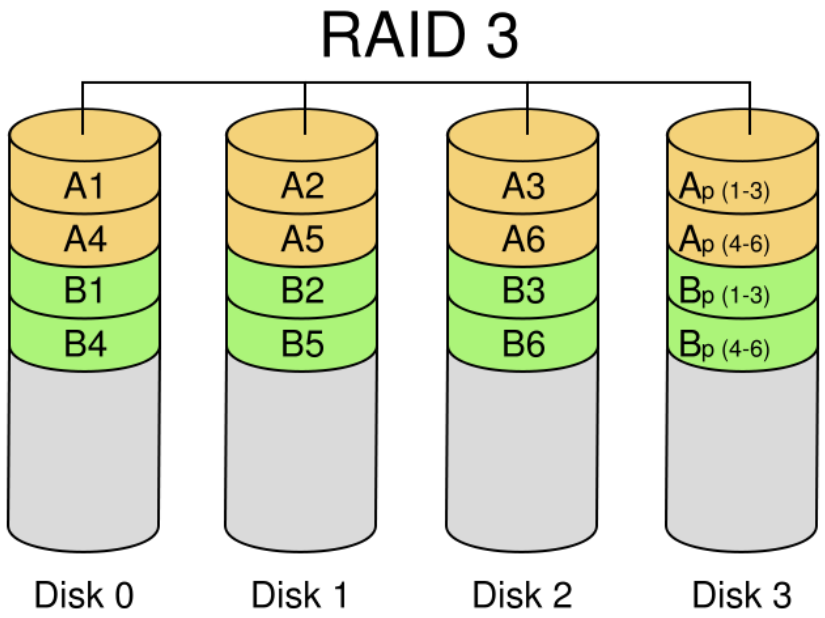
RAID 2

- Hamming Code ECC
- *Bit interleave*
- Kasutati *mainframe*-de ajastul



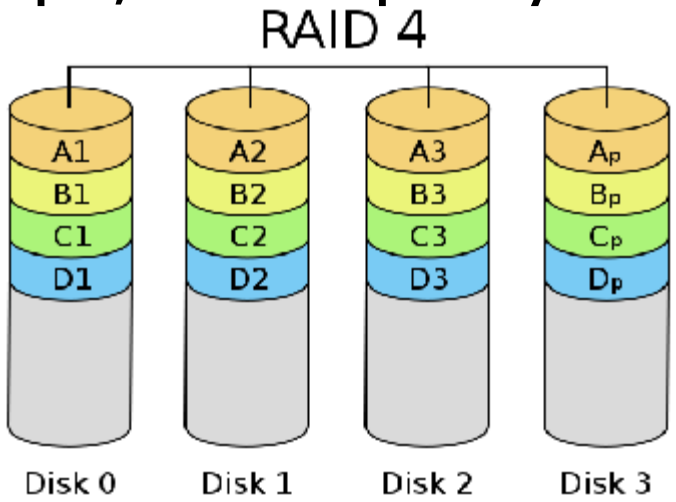
RAID 3

- Parallel transfer with parity
  - *Byte interleave*
  - 'parity' info eraldi kettal
  - Mahutavus: n-1
  - Kiirus
    - järjestikusel lugemisel või kirjutamisel suurepärane
    - ainult üks pöördumine korraga!
  - Andmed säilivad ühe ketta riknemisel, kiirus ei muutu!



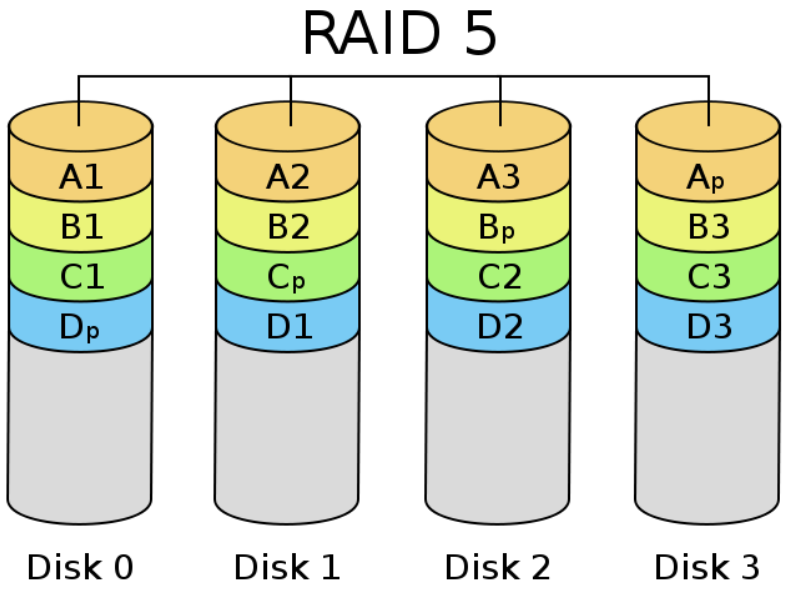
RAID 4

- Independent data disks with shared parity disk
  - striping nagu RAID0-gi plokikaupa, lisaks ‘parity’ info eraldi kettal
  - Mahutavus: n-1
  - Kiirus
    - lugemisel – suurepärane
    - kirjutamisel – hea
      - mitmikpöördusel jääb pudelikaelaks `parity´ ketas
  - Andmed säilivad ühe ketta riknemisel



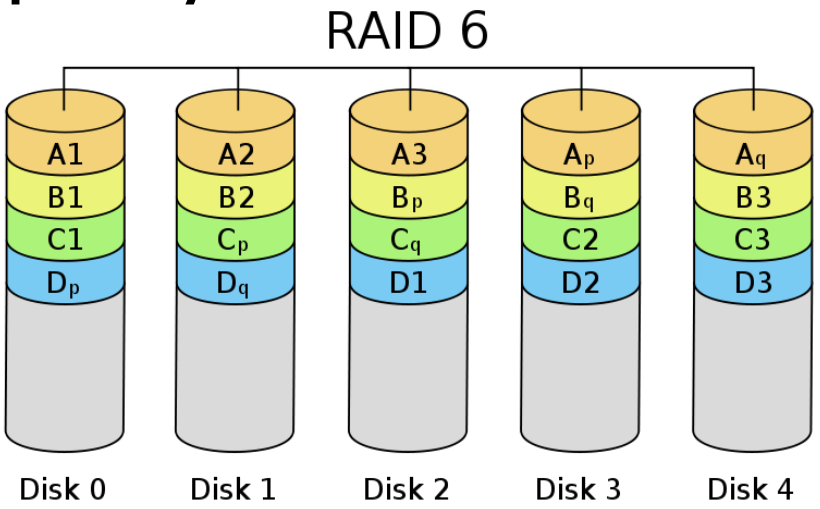
RAID 5

- Independent data disks with distributed parity blocks
  - ‘parity’ info on ketastel „laiali“
  - Mahutavus: n-1
  - Kiirus
    - lugemisel – suurepärane
    - kirjutamisel – hea...väga hea
  - Andmed säilivad ühe ketta riknemisel



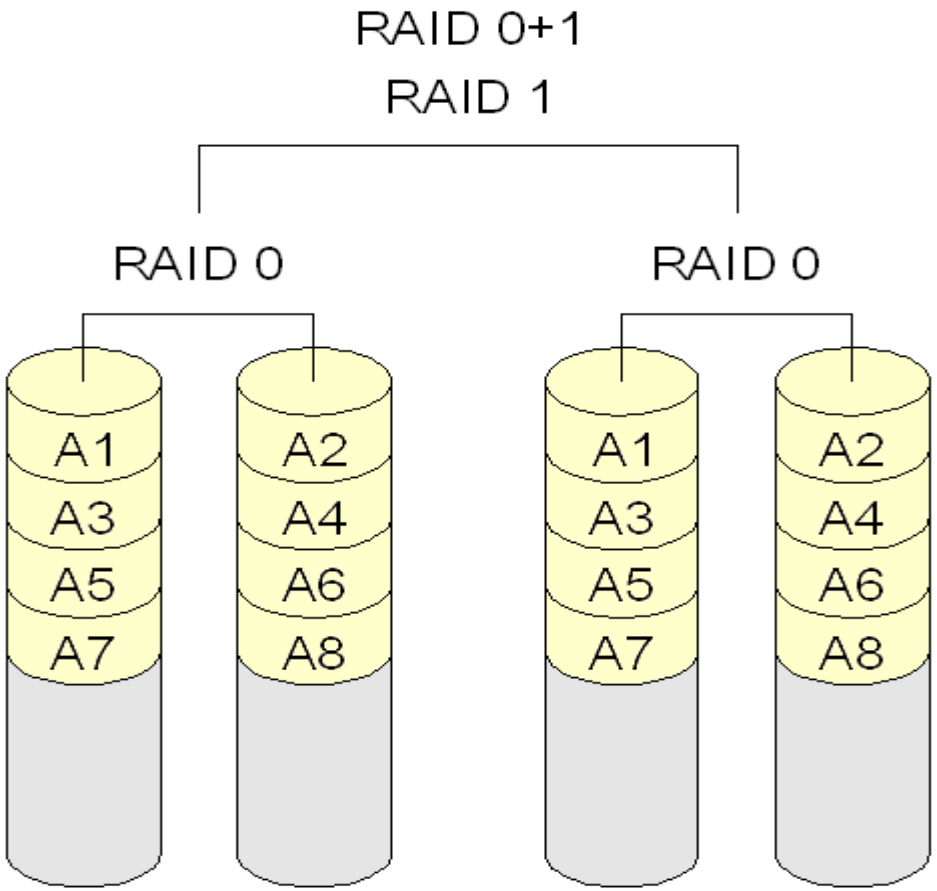
RAID 6

- Independent data disks with two independent distributed parity schemes
  - `parity`-infot 2 instantsi, muidu nagu RAID 5
  - Maht: n-2
  - Kiirus
    - lugemisel – suurepärane
    - kirjutamisel – hea
  - Andmed säilivad kuni kahe ketta riknemisel
    - NB! Teise ketta riknemist oodata EI soovita!



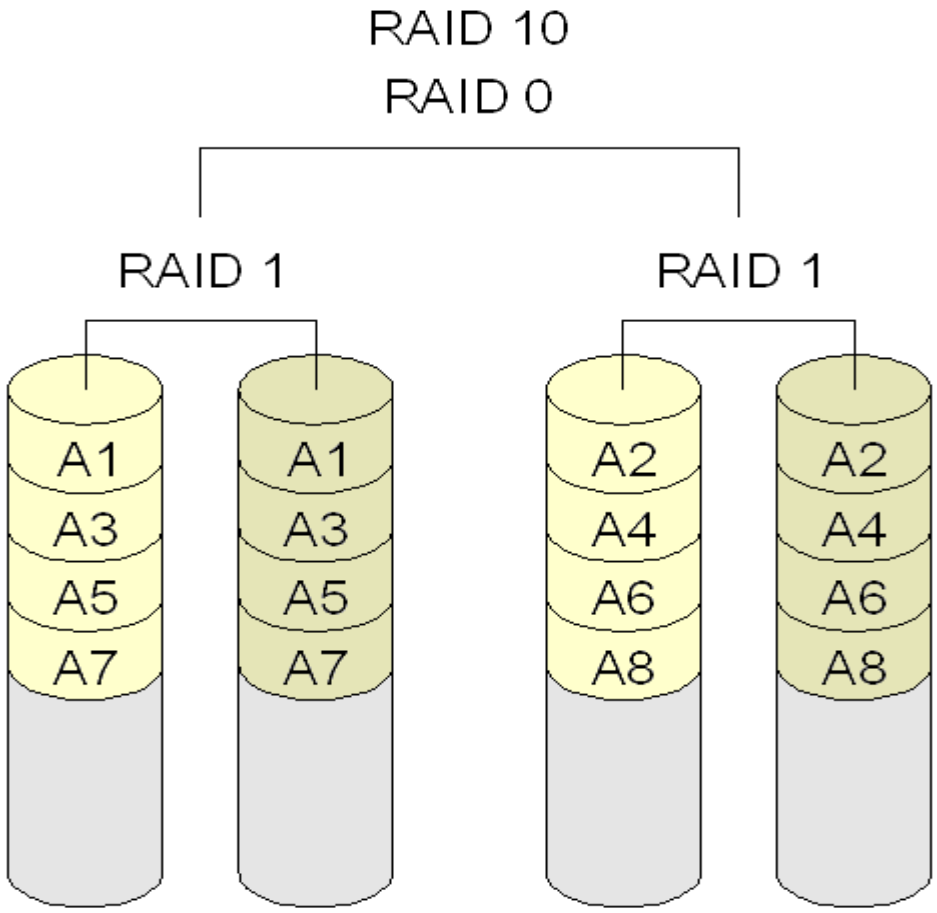
RAID

- RAID 0+1 mirrored array whose segments are RAID 0 arrays



RAID

- RAID 10 - striped array whose segments are RAID 1 arrays



RAID

- riistvaraline
  - ei koorma arvuti protsessorit
  - on OS-st sõltumatu
    - OS-le paistab köide *(volume, virtual disk)* kui tavaline füüsiline ketas
- tarkvaraline
  - odav – ei vaja eraldi riistvara
  - Linux-s mitu võimalust
    - md- seadmed (nt /dev/md123)
    - LVM VG võib koosneda mitmest PV-st RAID-na
    - failisüsteemid nagu ZFS või BTRFS oskavad sisemiselt

RAID ja failisüsteemid