Failisüsteemide realiseerimine

- Failisüsteemi struktuur
- Failisüsteemi realiseerimine
- Kataloogide realiseerimine
- Ruumi hõivamise meetodid
- Vaba ruumi haldus
- Efektiivsus ja jõudlus
- Taastumine vigadest
- Logivad failisüsteemid
- Näide: NFS

Failisüsteemi kihiline struktuur

- Rakendusprogrammid
- Loogiline failisüsteem – tegeleb metainfoga (kataloogistruktuur, faili kontrollplokid)
- Failide organiseerimise meetod – teab failide asukohtadest, loogilistest ja füüsilistest plokkidest, vabast ruumist
- Plokkseadmete tase – plokkide I/O haldus, puhverdamine
- Seadmedraiverid – tõlgivad üldisi I/O käske seadme käskudeks ja suhtlevad seadmega
- Seadmed

Mida üks failisüsteem kettal hoiab?

- (Partitsioonitabel – väljaspool konkreetset failisüsteemi)
- Alglaadeplokk (UFS: *boot block*, NTFS: *partition boot sector*) – opsüsteemi buutimiseks vajalik info
- Failisüsteemi juhtplokk – konkreetse failisüsteemi detailne info (plokkide arv, vabade plokkide arv ja asukoht, vabade failikirjete arv ja asukoht, . . .). UFS puhul *superblock*, NTFS puhul MFT (*Master File Table*)
- Kataloogistruktuur – kataloogipuu hoidmiseks
- Failide kontrollplokid (FCB – *File Control Block*) – Detailid iga konkreetse faili kohta. UFS puhul i-kirje, NTFS puhul asub MFT sees tabelis

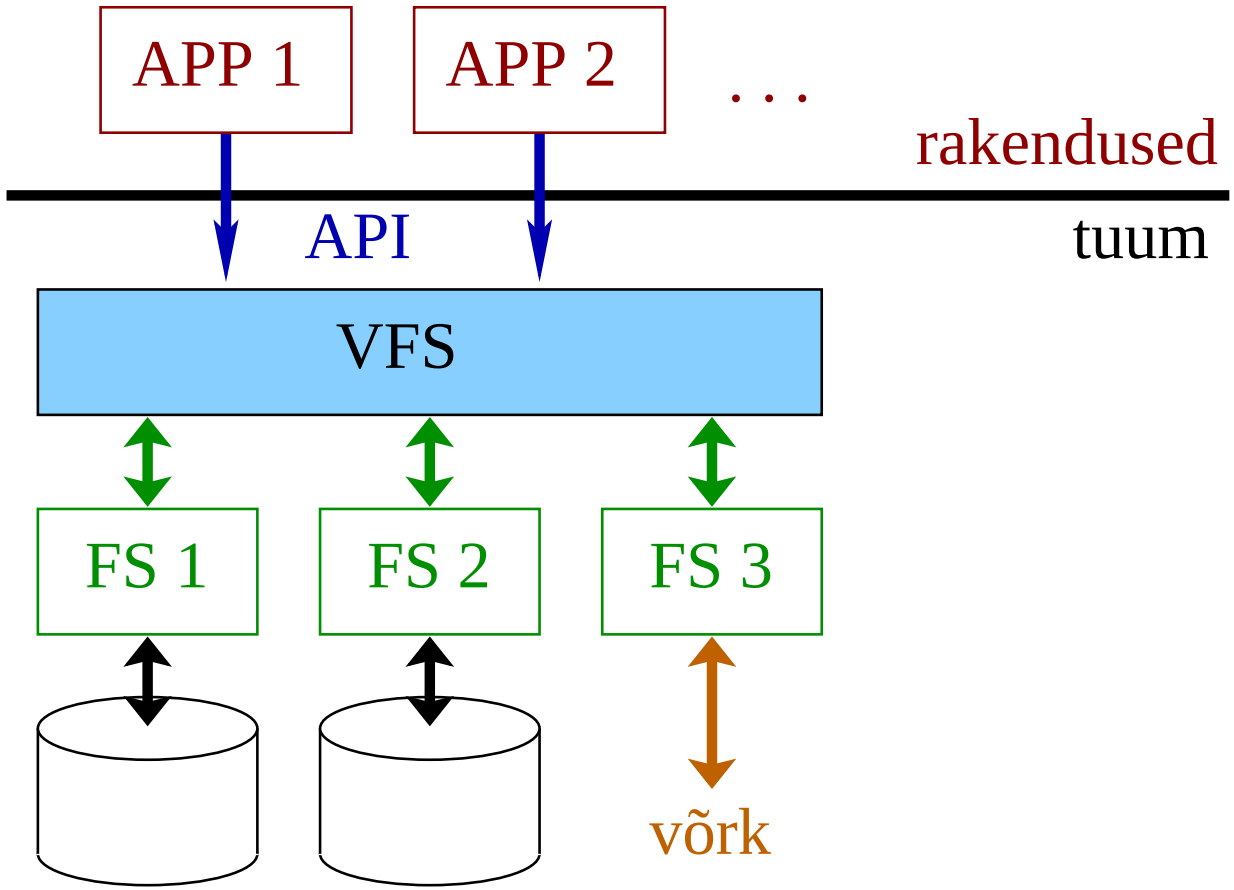
Mida failisüsteem mälus hoiab

- (Monteerimispunktide tabel)
- Monteeritud failisüsteemide tabel
- Viimati kasutatud kataloogistruktuuri kirjed (ja muu metainfo) puhverdamise mõttes
- Puhverdatud andmeplokid
- Süsteemne avatud failide tabel – avatud failide FCB-de koopiad
- Iga protsessi avatud failide tabel
  - UNIX: failideskriptorid (*file descriptor*)
  - Win32: failipidemed (*file handle*)

Virtuaalne failisüsteem

- Objekt-orienteeritud lähenemine paljude failisüsteemide realiseerimisele samas opsüsteemis
- VFS laseb sama süsteemifunktsioonide liidest kasutada paljudel erinevatel failisüsteemide tüüpidel
- Rakendusporgrammid liidestuvad VFS-ga, mitte konkreetse failisüsteemiga
- VFS pakub konkreetsetele failisüsteemidele omakorda madalama taseme liidese, mida need kasutavad oma info süsteemile edastamiseks
- Konkreetseid failisüsteemi tüüpe võib vaadelda kui alamklasse, mis realiseerivad sama liidese
- VFS realiseerib ära selle osa funktsionaalsusest, mis on failisüsteemidele ühine ja antud opsüsteemis kohustuslik

Virtuaalne failisüsteem

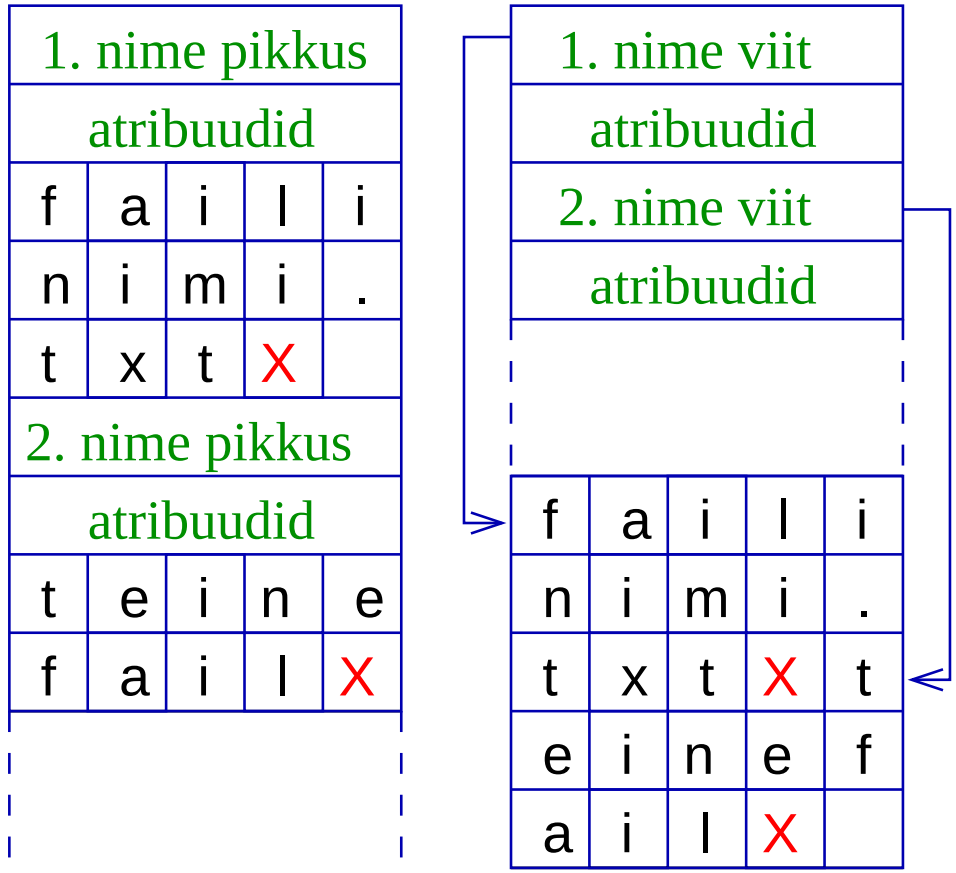


Kataloogide realiseerimine

- Lineaarse nimekirjana
  - Lihtne realiseerida
  - Aeglane suure failide arvu juures
- Paisktabelina
  - Otsing on kiirem
  - Fikseeritud suurus; kollisioonid
- Puuna (tihti näiteks B-puud)
  - Otsing on kiire
  - Efektiivne
  - Keeruline programmeerida
- Kombineeritud (puu + paisksalvestus)

Kataloogide realiseerimine

- Muutuva pikkusega failinimede käsitlemine



Ruumi hõivamine

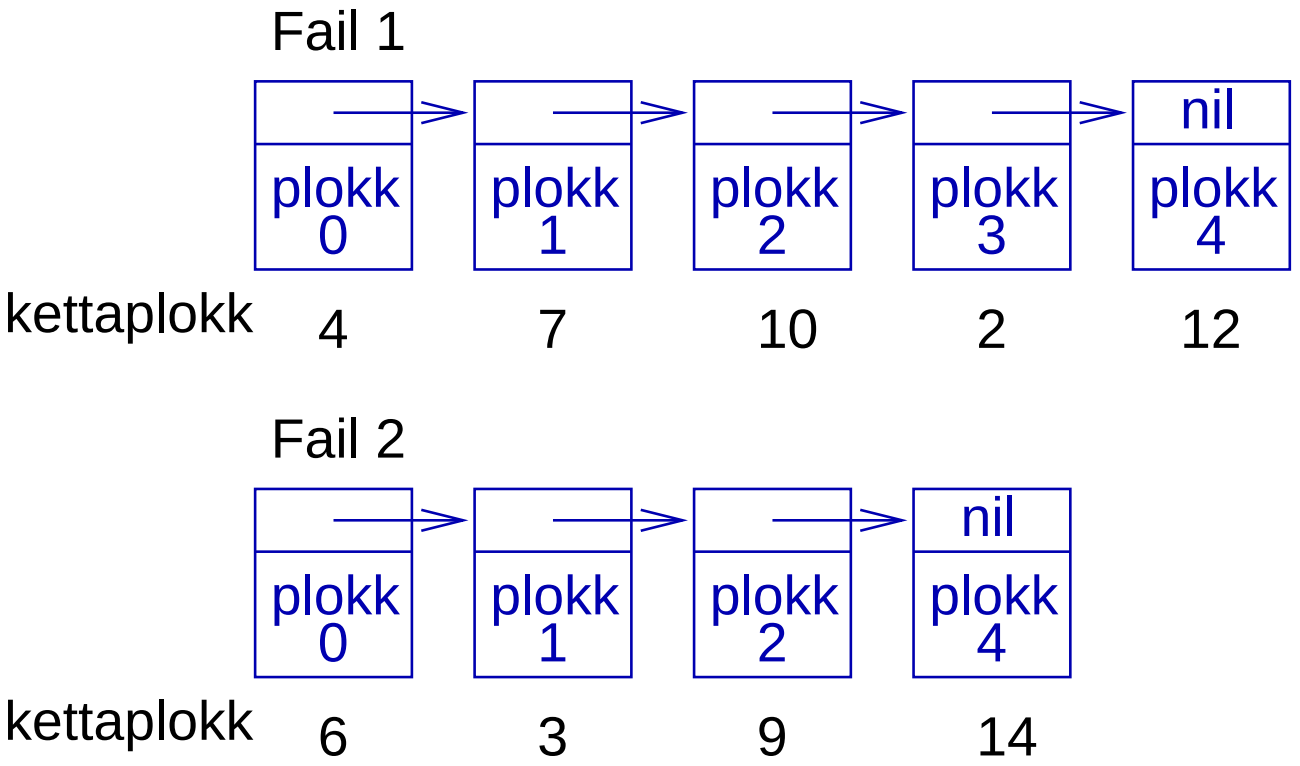
- Kuidas konkreetsele failile kettaplokke jagada?
- Kolm põhilist meetodit:
  - Pidev tükk
  - Lingitud paigutus
  - Indekseeritud paigutus

Pideva tükina hõivamine

- Iga fail katab mingi pideva plokkide jada kettal
- Lihtne – ainult esimese ploki number ja faili pikkus plokkides on vaja meelde jätta
- Otsepöördus suvalise ploki poole
- Raiskab ruumi (dünaamiline ruumihalduse probleem nagu mälu juures)
- Fail ei saa kasvada (või on kasvamine keeruline / aeglane)
- Mitmed uuemad failisüsteemid (XFS, JFS, Veritas File System, QFS) kasutavad pideva paigutuse modifitseeritud varianti:
  - Kettaruumi jagatakse ulatuste (*extent*) kaupa. See on üks sidus ala. Fail koosneb ühest või mitmes ulatusest.

Lingitud paigutus

- FCB-s on esimese ploki number
- Iga ploki sisse pannakse järgmise ploki number
- Tekib ahel (lõpetab spetsiaalne number – *nil*)

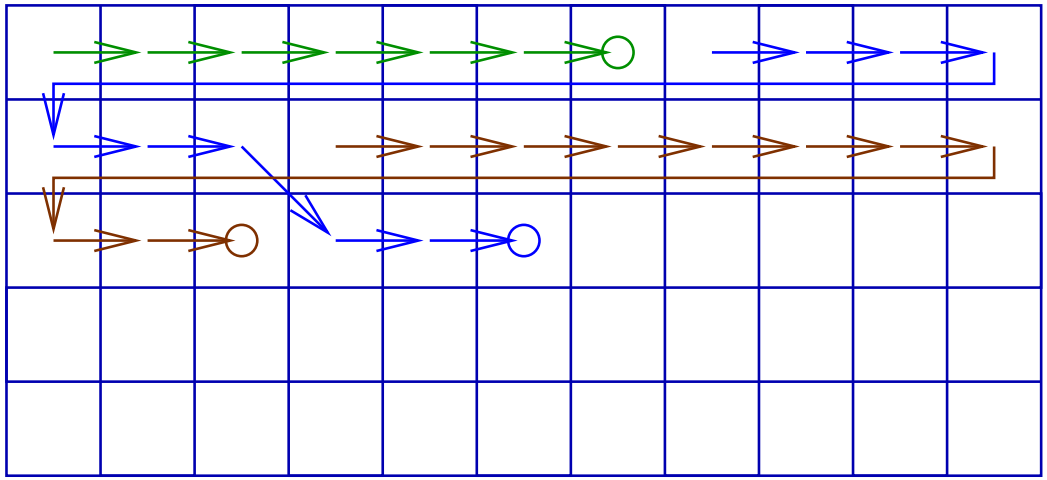


Lingitud paigutus

- Vaba ruumi ei raisata (pole fikseeritud pikkusi)
- Lihtne realiseerida
- Otsepöördust pole – ahel tuleb iga kord algusest läbida
- Klastrid – hõivame ruumi mitme sektori kaupa
- Õrn vigade suhtes (ahel läheb kergesti katki)

Lingitud paigutus – FAT

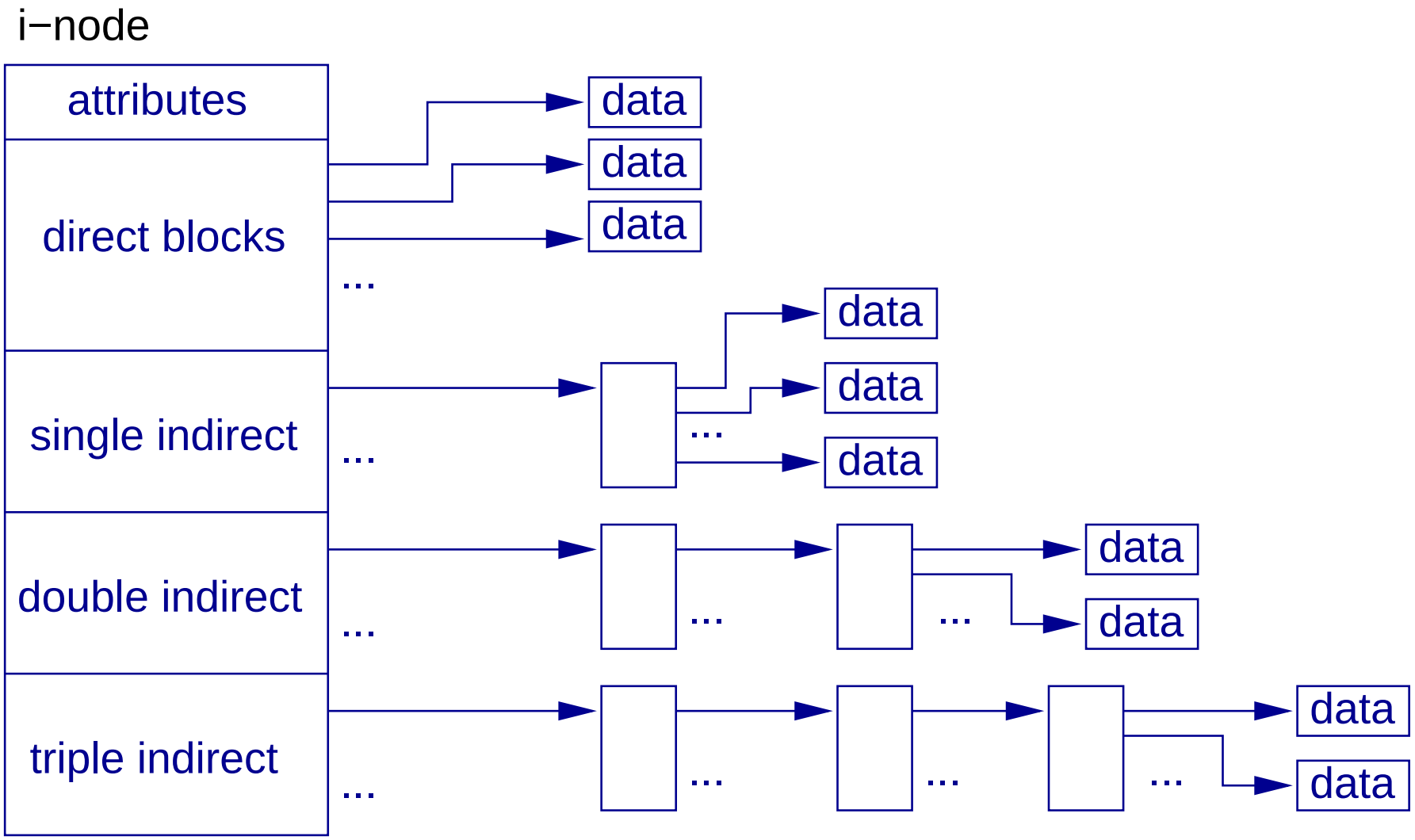
- FAT (*File Allocation Table*) – viitade tabel on eraldi kettaosas
- FAT tabel on reeglina puhverdatud mällu
  - Liigse positsioneerimise vältimiseks
  - Saame ka otsepöörduse
- Ruumi hõivatakse klastrite kaupa
- Kasutusel DOS-is, eemaldatavatel seadmetel



Indekseeritud paigutus

- Tehtud efektiivsema otsepöörduse jaoks
- Kõik plokkide viidad tuuakse kokku ühte kohta
- Iga FCB-ga seotakse viitade plokk – viidad kõigile selle faili plokkidele
- Raiskab ruumi viitadele tervete plokkide kaupa, aga välist fragmenteerumist pole
- Lingitud indekseerimine: ahel indeksiplokkidest
- Mitmetasemeline indeks: indeksiplokkidest tehakse puu
- Kombineeritud skeem: natuke plokke viidatakse otse, natuke 1-tasemelise puuna, edasi 2-tasemelise puuna ja ülejäänud 3-tasemelise puuna (UFS)
- Aukudega failid

Kombineeritud indekseeritud paigutus



Vaba ruumi haldus

- Kuidagi tuleb meeles hoida, missugused kettaplokid on vabad
- Bitivektoriga: iga ploki kohta on bitt (1, kui plokk on vaba)
  - Bitte on niipalju kui failisüsteemis kasutatavaid andmeplokke
  - Otsimisel leitakse esimene 0-st erinev sõna ja selle seest esimene nullist erinev bitt
  - Võtab suhteliselt palju ruumi
  - Pidevaid alasid lihtne leida
- Ahelana
  - Ahel on plokkidest endist või eraldiseisev (nagu FATil)
  - Sidusaid alasid raske leida
- Grupeerimine
- Loendamine

Efektiivsus ja jõudlus

- Failisüsteemi efektiivsus ja jõudlus sõltuvad paljuski andmete paigutusest kettal ja kataloogialgoritmidest
- Näide: UFS ja silindrigrupid ning i-kirjete laiali jaotamine
- Näide: faili viimase kasutuse aja (*atime*) pidevalt kettale kirjutamine või kirjutamata jätmine
- Kettapuhvrid (*disk cache*) – mäluosa, mis on eraldatud kettal olevate sagedasti kasutatavate andmete puhverdamiseks
- *Free-behind* ja *read-ahead* – tehnikad järjestikpöörduse optimeerimiseks
- `madvise()` süsteemifunktsioon
- Sünkroonsed ja asünkroonsed kettaoperatsioonid
- Mäluketas (*RAM disk*)

Lehekülgede puhverdamine

- I.k. *page cache*
- Esimene vahemälu on kettaseadmes – vähemalt 1 rada tuleb puhverdada
- Sellest jääb väheks, ketta andmeid tuleb mälus ka puhverdada (puhverdame kettaplokkide tasemel)
- Samm edasi: hoiame infot mälus failidega seotult – iga mälus olev lehekülg on mingi faili seest või swapist
- Unifitseeritud virtuaalmälu – sama lehekülgede puhvrit kasutatakse nii failide lehekülgede kui protsesside andmete jaoks
- Topeltpuhverdamine vs unifitseeritud vahemälu
- Kuidas jagada mälu kettapuhvrite ja protsesside lehekülgede vahel?

Taastumine vigadest

- Osasid andmestruktuure puhverdatakse mälus kiiruse huvides
- Aeg-ajalt tuleb seda kettale kirjutada
- Vahel läheb vool enne ära või juhtub muu jama
- Kirjutamisoperatsiooni pooleli jäämisel võivad erinevad struktuurid kettal omavahel sünkrost väljas olla
- Vaja parandada (enne järgmist monteerimist näiteks)
- Alati ei saa parandada $\Rightarrow$ varundamine
- Inkrementaalne varundamine

Vigade parandamine

- Failisüsteemi kooskõla kontrolliks on süsteemsed utiliidid
  - UNIX: fsck (fsck.*failisüsteem*)
  - DOS: CHKDSK
  - Windows: ScanDisk
- Failide/kataloogistruktuuri kooskõla kontrollimine
  - Näiteks kas failile viitavate linkide arv on sama, mis loenduris?
- Plokkide kooskõla kontrollimine
  - Iga plokk peab olema kasutusel täpselt ühe faili või kataloogi poolt või siis vabade plokkide nimekirjas
- Muu metainfo kooskõla kontrollimine

Varundamise meetodid

- Plokkseadme tasemel
  - Varundatakse üksteise järel plokkseadme kõigi andmeplokkide sisu
  - Lihtne realiseerida, kiire
  - Vigased plokid on probleemiks
  - Ebaefektiivne lindi kasutus (varundatakse ka vabad plokid)
  - Konsistentse seisu saamise vajadus
- Failide tasemel
  - Varundatakse rekursiivselt mingi kataloogi alla jääv puu
  - Toetab inkrementaalset varundamist
  - Hoiab ruumi kokku
  - Võib probleeme tulla spetsiifilisemate failiatribuutidega

Logivad failisüsteemid

- Ingl.k. *(log-structured | log-based | logging | journaling | transaction-oriented) file systems*
- Idee võetud andmebaasidelt: kuidas hoida kettal igal ajahetkel konsistentne seis
- Näited: NTFS, Veritas VxFS, Solarise UFS, Linuxi Ext3
- Logi võib olla andmetega samal plokkseadmel või kiiruse huvides teisel seadmel
- Mitut sorti logimist:
  - Ainult metaandmete logimine
  - Metaandmete muutuste järjestamine
  - Täislogimine
  - *Softupdates* – natuke teistmoodi lähenemine

Logivate failisüsteemide tööpõhimõte

- Transaktsioon – mingi fikseeritud metainfo muutus või fikseeritud hulk andmeplokkide muutusi
- Peetakse logi transaktsioonide kohta – kõik transaktsioonid kirjutatakse järjest logisse
- Taustal mängitakse logi maha päris failisüsteemi peal
- Iga transaktsiooni pärisfailisüsteemi kirjutamise järel kustutatakse see transaktsioon logist
- Crashi järgsel monteerimisel mängitakse logist maha seal olevad terved transaktsioonid ja keritakse tagasi poolikud
- Modifikatsioon: faasipuudel põhinevad failisüsteemid, "tantsivad puud"

NFS – *Network File System*

- Originaalselt Suni (ONC+) võrgufailisüsteem, praktikas *de facto* standard UNIXite vahel failide jagamiseks
- SunRPC (ONC RPC) üle UDP (LAN puhul) või TCP (WAN puhul)
- Klient-server mudel
- Klient omab failisüsteemidraiverit, mis oskab suhelda serveritega
- Server ekspordib oma failisüsteemist mingeid osi, klient monteerib selle oma failisüsteemi mingisse kataloogi
- Rakendusprogrammidele läbipaistev
- Monteerimine pole läbipaistev, serveri nimi (nimed) ja asukoht serveris on vaja ette anda (VFS tabel)
- Eeldab samu kasutajakontosid kõigis kasutatavates masinates

NFS: protokollid

- Monteerimise protokoll – annab esialgse failipideme jagatud kataloogi tipu kohta, kui see on kliendile lubatud
- Failiedastuse protokoll – teades failipidet, saab teha failioperatsioone ja pidemeid alamkataloogidele
- Muud protokollid – lukustamine, quota, . . .
- Automaatne monteerimine (kodukataloogid; naabermasinad /net alla; . . .)

NFS: failide jagamise protokoll

- Saab RPC kaudu süsteemifunktsioonidele analoogseid protseduuriväljakutseid
- Faile eristatakse failipidemete järgi
- Olekuvaba (*stateless*) failiserver
- Kataloogist saab alamkataloogide ja failide kohta failipidemeid edasisteks operatsioonideks
- NFS v2 on sünkroonne, v3 laseb eriliidese järgi kasutada ka asünkroonset liidest
- NFS v4 on ühend NFS v3 ja Windowsi SMB-st, ühendades mõlema omadusi (olekut säilitav)