Failisüsteemide realiseerimine

- Failisüsteemi struktuur
- Failisüsteemi realiseerimine
- Kataloogide realiseerimine
- Ruumi hõivamise meetodid
- Vaba ruumi haldus
- Efektiivsus ja jõudlus
- Taastumine vigadest
- Logivad failisüsteemid
- Näide: NFS
- Näide: Ext2
- Näide: NTFS

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 ja järjekorrad
- 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

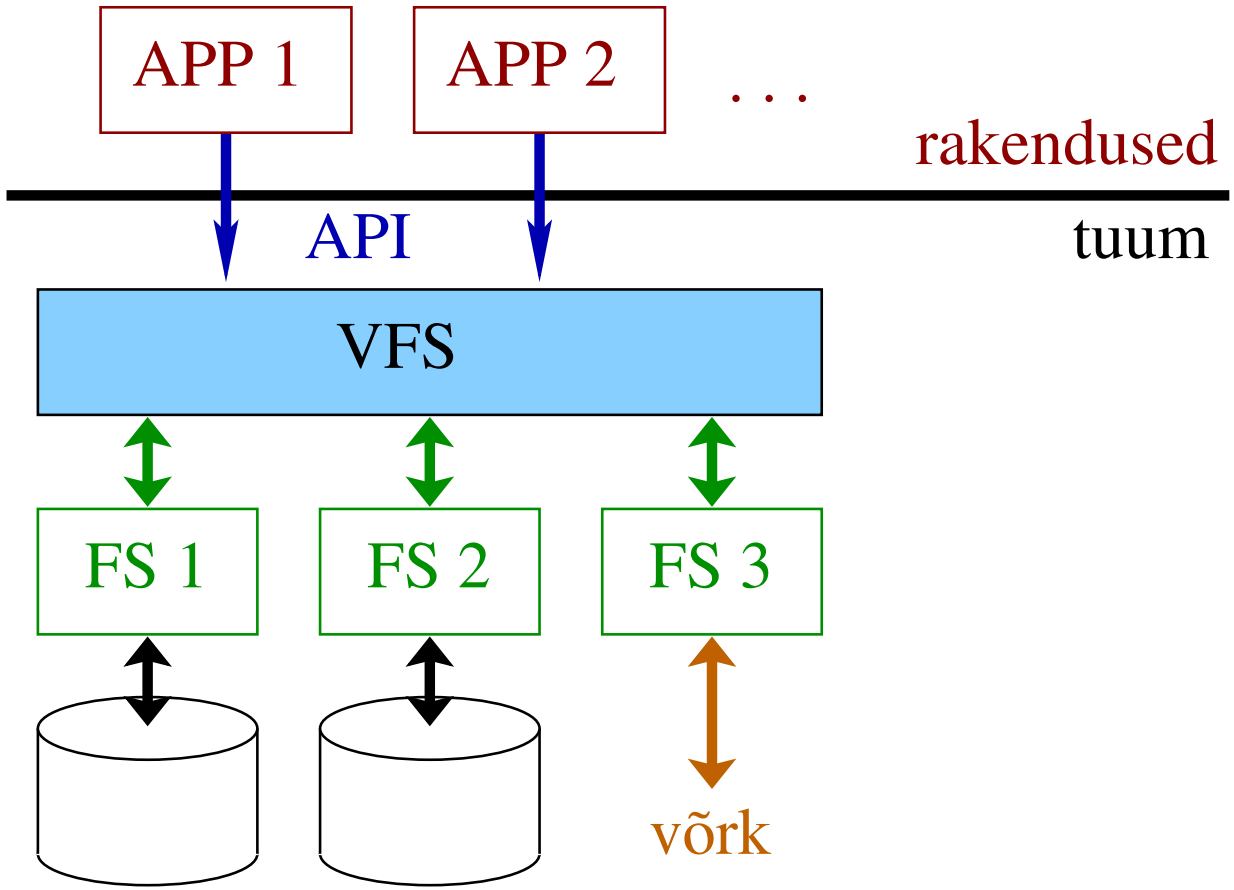
Mida failisüsteem mälus hoiab

- (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

- Objektorienteeritud lähenemine paljude failisüsteemide realiseerimisele samas opsüsteemis
- VFS laseb sama süsteemifunktsioonide liidest kasutada paljudel erinevatel failisüsteemide tüüpidel
- Rakendusprogrammid 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

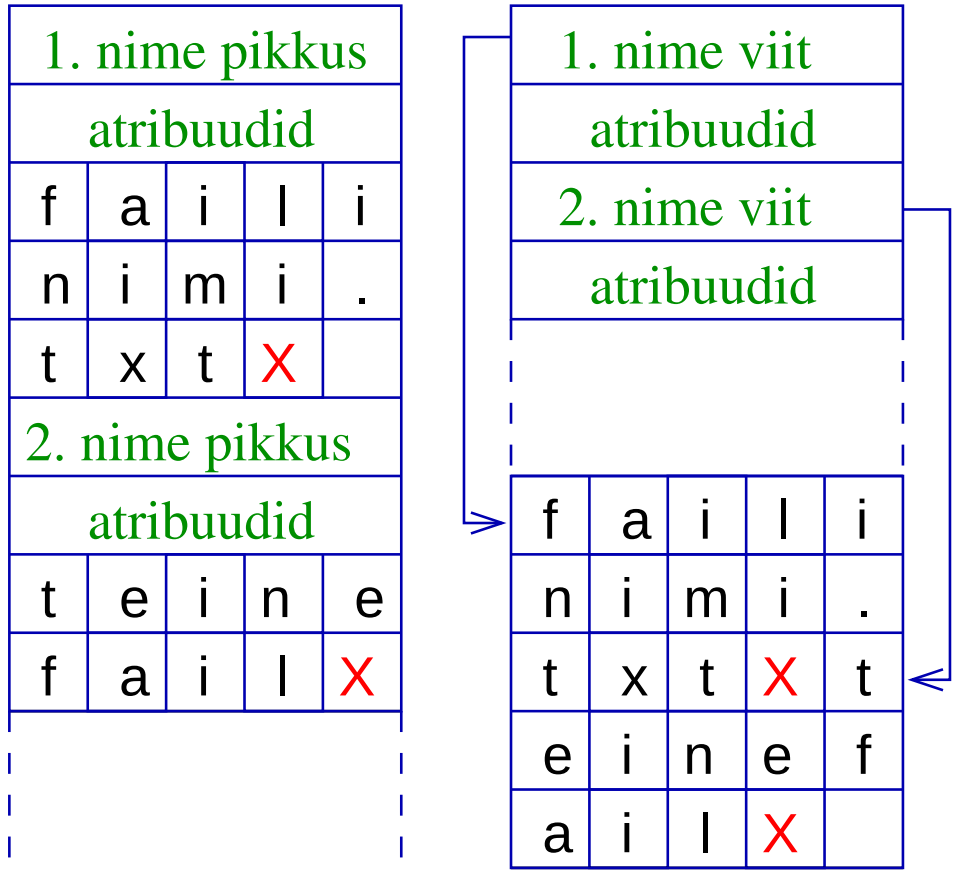


Kataloogide realiseerimine

- 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

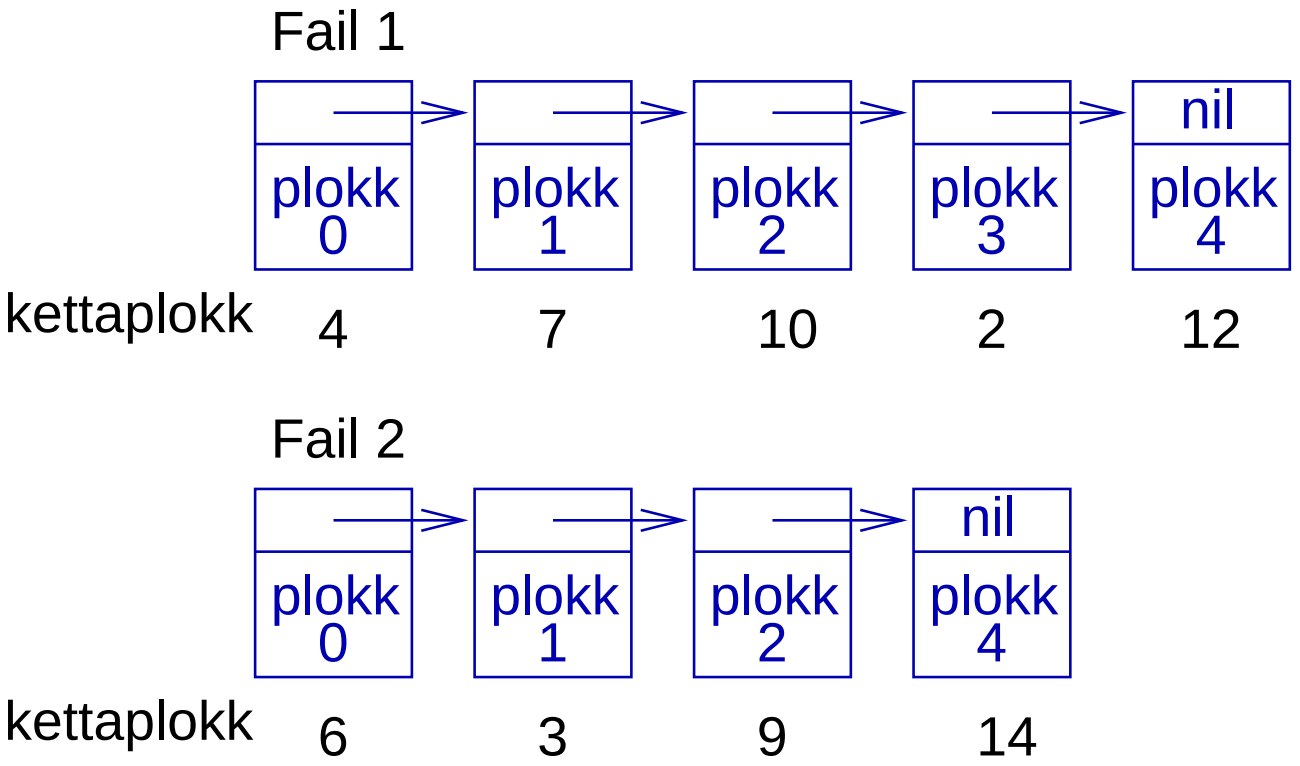
- Kuidas konkreetsele failile kettaplokke jagada?
- Kolm põhilist meetodit:
  - Pidev tükk
  - Lingitud paigutus
  - Indekseeritud paigutus

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, Ext4) 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*)

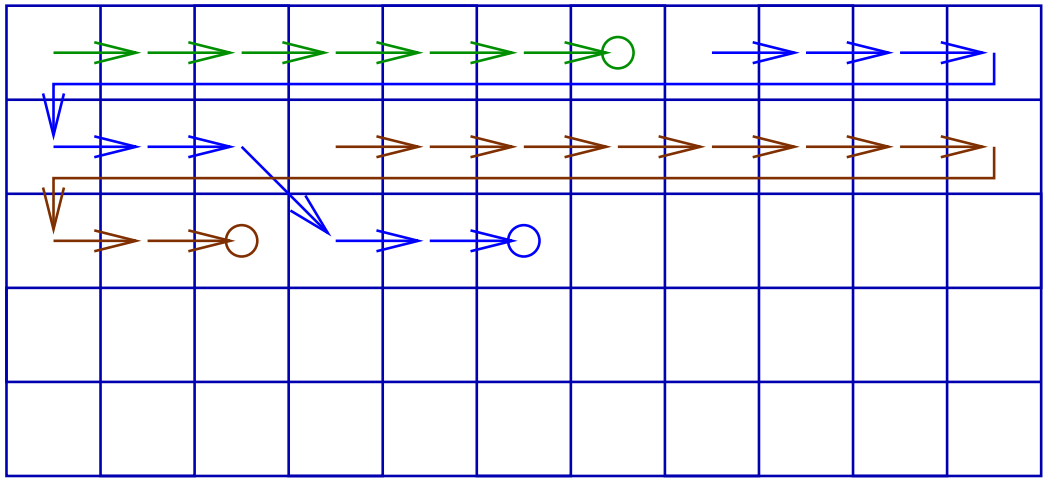


Lingitud paigutus

- 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

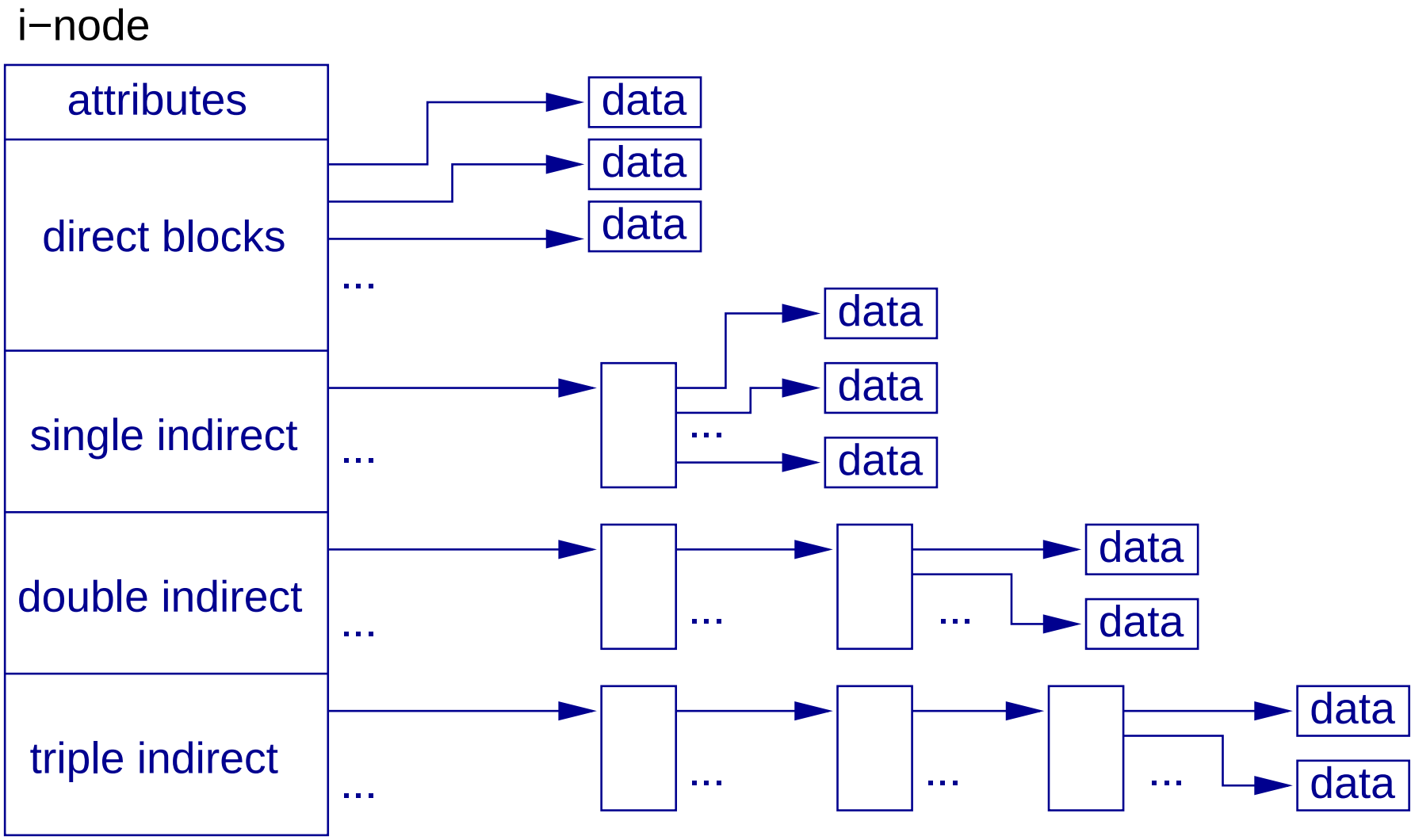
- FAT (*File Allocation Table*) — viitade tabel on eraldi kettaosas
- FAT tabel on reeglina puhverdatud mällu
  - Liigse positsioneerimise vältimiseks
  - Saame ka otsepöörduse
- Ruumi hõivatakse klastrite kaupa (4kiB-64kiB)
- Kasutusel DOS-is, eemaldatavatel seadmetel
- FAT12, FAT16, FAT32, TFAT, ExFAT, pikad failinimed



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 on lihtne leida
- Ahelana
  - Ahel on plokkidest endist või eraldiseisev (nagu FATil)
  - Sidusaid alasid on 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
- `fadvise()` ja `madvise()` süsteemifunktsioonid
- 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 need 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

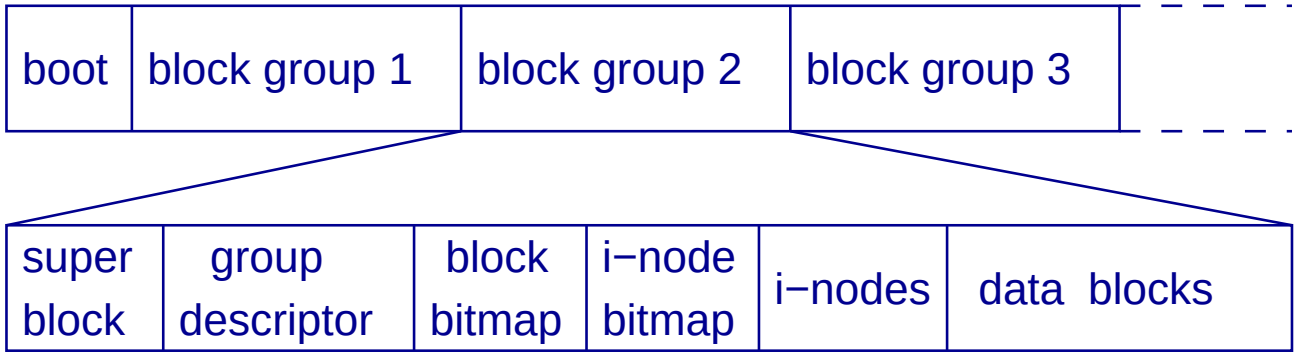
- Saab RPC kaudu süsteemifunktsioonidele analoogseid protseduuriväljakutseid
- Faile eristatakse failipidemete järgi
- Olekuvaba (*stateless*) failiserver
- Kataloogist saab alamkataloogide ja failide kohta failipidemeid edasisteks operatsioonideks
- NFS v2 on sünkroonne, v3 laseb eriliidese järgi kasutada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

Näide: Ext2 perekond

- Oli pikka aega Linuxi põhiline failisüsteem
- Järglaseks oli Ext3
  - Andmeformaadid on samad
  - Spetsiaalne i-kirje transaktsioonide logiga
  - Kataloogisisesed B-puud
  - Vähem superploki koopiaid
- Tänapäevaseks asenduseks on Ext4
  - Ulatustega paigutus
  - Suuremad mahud
  - Hilistatud paigutus (*delayed allocation*)
  - Täpsemad ajatemplid
  - Kiirem *fsck*

Näide: Ext2 failisüsteem

- Ketas (plokkseade) on jagatud plokirühmadeks (*block group*)
- Failisüsteemi paigutus kettal:



- Superplokk — ext2 metainfo (plokisuurus, plokkide arv, . . .)
- Rühmadeskriptor — bititabelite asukoht, vabade plokkide ja i-kirjete arv, rühmas olevate kataloogide arv

Näide: Ext2 failisüsteem

- Igale objektile failisüsteemis vastab i-kirje
- I-kirje pikkus on 128 baiti
  - Ext3 toetab ka 256-baidiseid i-kirjeid, ext4 256+
- I-kirjes on metainfo objekti kohta (v.a. objekti nimi) ja ning viidad andmeplokkidele
- Objekti metainfo sisaldab
  - Objekti tüüpi (fail, kataloog, nimeviit, . . .) ja õigusi
  - Omanikku ja gruppi
  - Faili suurust baitides
  - Viimase lugemise/kirjutamise/i-kirje muutmise aegu
  - I-kirjele viitavate viitade arvu
  - . . .

Näide: Ext2 failisüsteem

- I-kirjes on kokku 15 viita andmeplokkidele
  - 12 otseviita ja 3 viita kaudset viita (vastavalt ühe-, kahe- ja kolmetasemeliseks indekseerimiseks)
  - Kuni 60-märgine nimeviit ei kasuta andmeplokke, vaid viitade välju
- Kõik i-kirjed tekitatakse failisüsteemi loomisel, hiljem nende arvu muuta ei saa
- On spetsiaalseid programme, mis monteerimata failisüsteeme suurendada ja vähendada oskavad, ka ext2 jaoks (resize2fs)

Näide: Ext2 failisüsteem

- Ploki suurus 1 — 8 kB
- Faili suurust piirab viitade maksimaalne arv i-kirjes ja 32-bitine plokkide arv

Plokisuurus	max failipikkus	põhjus
1 kB	16 GB	viitade arv
2 kB	256 GB	viitade arv
4 kB	16 TB	32-bitine plokkide arv

- 32-bitine 512-baidiste sektorite arv piirab varasemates Linuxi versioonides kõigi plokkseadmete suuruse 1 TB peale
- Veel varasem piirang piiras 32-bitistel arhitektuuridel kõigis failisüsteemides faili maksimaalse suuruse 2G-ga

Näide: NTFS

- MFT (*Master File Table*) — kogu ülejäänud ketta indeks
- Ruumi jagatakse klastri kaupa, klaster 0.5 – 4 KB (32/64 KB)
- Logiv failisüsteem
- Kataloogid B-puudena
- Indekseeritud paigutus, indekseeritakse ulatusi
- Suvaline atribuutide hulk failidel
- Residentsed ja mitteresidentsed atribuudid
- Failinimed Unicode, tõstutundlikkus sõltub liidesest
- *Reparse point* — nimeviited, failisüsteemide monteerimine, . . .
- Harudega failid
- Pakkimine, krüptimine, kettakvoodid, ACL

NTFS erifailid

Metaandmed	Failinimi	Kirjeldus
Master File Table	$MFT	Sisaldab iseennast
Master File Table 2	$MFTMirr	MFT 16 esimese sektori koopia
Log File	$LogFile	transaktsioonilogi
Volume Descriptor	$Volume	superplokk
Attribute Def. Table	$AttrDef	Failiatribuutide kirjeldused
Root Directory	.	Viit juurkataloogile
Cluster Alloc. Bitmap	$Bitmap	Vaba ruumi bitikaart
Volume Boot Code	$Boot	Koopia bootsektori koodist
Bad Cluster File	$BadClus	Vigaste klastrite nimekiri
Quota Table	$Quota	Kettakvootide tabel
Upper Case Table	$UpCase	Failinimede tõlketabel