Failisüsteemide realiseerimine

Ülevaade

- Failisüsteemi struktuur
- Kataloogide realiseerimine
- Ruumi hõivamise meetodid
- Vaba ruumi haldus
- Efektiivsus ja jõudlus
- Taastumine vigadest
- Logivad failisüsteemid
- NFS
- Näited: MS-DOSi ja Ext2 failisüsteem

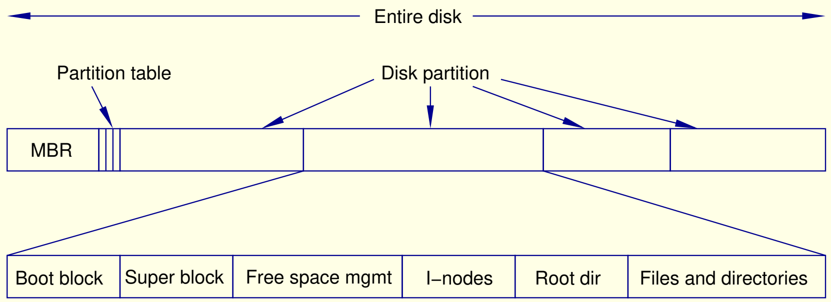
Failisüsteemi kihiline struktuur

- 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 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

Failisüsteemi võimalik paigutus kettal



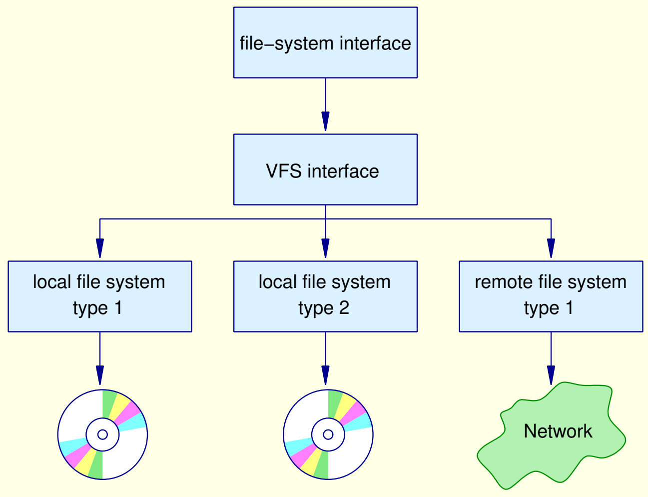
Mida failisüsteem mälus hoiab?

- (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 süsteemifuktsioonide liidest kasutada paljudel erinevatel failisüsteemide tüüpidel
- Rakendusporgrammid liidestuvad VFS-ga, mitte konkreetse failisüsteemiga
- Konkreetsetele failisüsteemidele pakub VFS 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

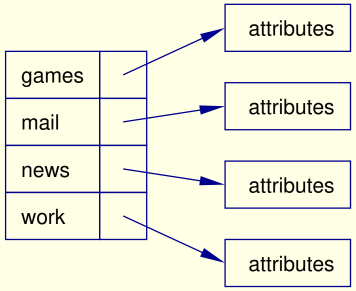
Virtuaalne failisüsteem



Kataloogide realiseerimine

- Kataloog on nimekiri failinimedest koos viitadega andmeplokkidele või FCB-dele
- Fikseeritud suurusega kataloogikirjed

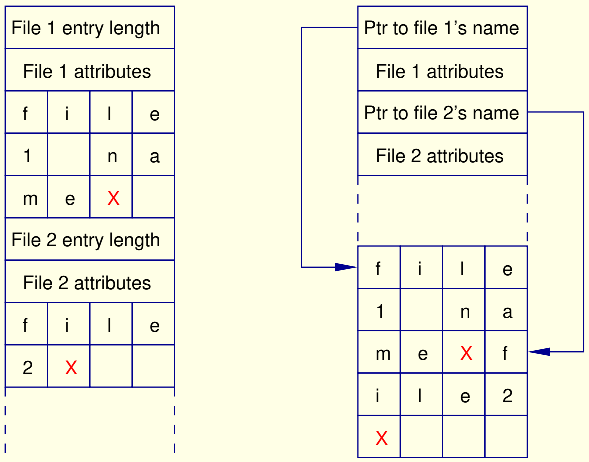
games	attributes
mail	attributes
news	attributes
work	attributes



- Failinimede pikkus fikseeritud (ja reeglina suhteliselt väike)

Kataloogide realiseerimine

- Pikkade failinimede käsitlemine



Kataloogide realiseerimine

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

Ruumi hõivamine

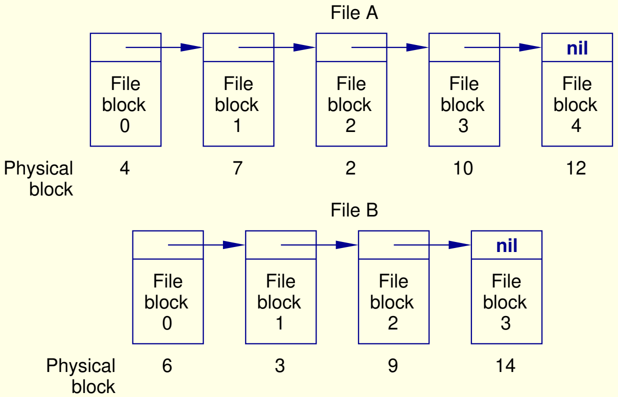
- Kuidas konkreetsele failile kettaplokke jagada?
- Kolm põhilist meetodit:
  - Pidev tükk
  - Lingitud paigutus
  - Indekseeritud paigutus

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 (Veritas File System näiteks) 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*)

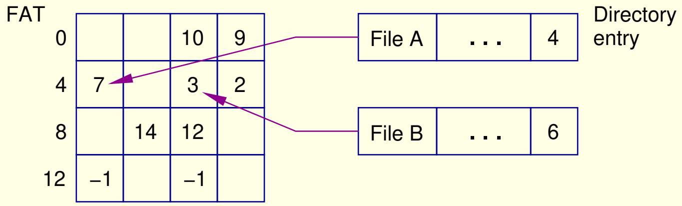


Lingitud paigutus

- Vaba ruumi ei raisata (pole fikseeritud pikkusi)
- Lihtne realiseerida
- Otsepöördust pole — ahel tuleb iga kord algusest läbida
- Õrn vigade suhtes (ahel läheb kergesti katki)
- Plokist kuluvad mõned baidid viida esitamiseks
  - Faili hoidmiseks kulub natuke rohkem ruumi
  - Klastrid
  - Plokis oleva info suurus pole kahe aste

Lingitud paigutus FAT-iga

- FAT (*File Allocation Table*) — viitade tabel on eraldi kettaosas



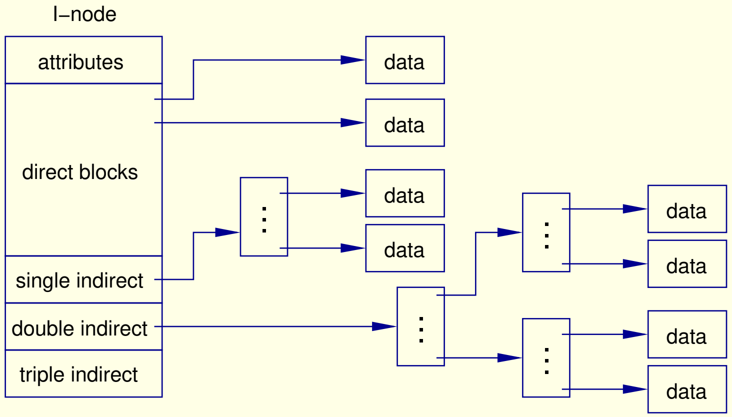
- FAT on reeglina puhverdatud mällu
  - Vähendab ketta lugemispea liigset ümberpositsioneerimist
  - Kiirendab otsepöördumist
- Kasutusel näiteks MS-DOS ja OS/2 failisüsteemides

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

Indekseeritud paigutus

- Kombineeritud skeem: natuke plokke viidatakse otse, natuke 1-tasemelise puuna, edasi 2-tasemelise puuna ja ülejäänud 3-tasemelise puuna (UFS)



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
- Kettapuhvrid (*disk cache*) — mäluosa, mis on eraldatud kettal olevate sagedasti kasutatavate andmete puhverdamiseks
- *Free-behind* ja *read-ahead* — tehnikad järjestikpöörduse optimeerimiseks
- Sünkroonsed ja asünkroonsed kettaoperatsioonid
- Māluketas (*RAM disk*)

Leheküilgede puhverdamine

- Ingl.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 sünkroonist väljas olla
- Vaja parandada (enne järgmist monteerimist näiteks)
- Alati ei saa parandada $\Rightarrow$ varundamine
- Inkrementaalne varundamine

Taastumine vigadest

- Failisüsteemi kooskõla kontrollimiseks süsteemsed utiliidid
  - fsck (Unix), ScanDisck (Windows)
- Failide / kataloogistruktuuri kooskõla kontrollimine
  - Näit. kas failile viitavate linkide arv on sama mis loenduris
- Plokkide kooskõla kontrollimine
  - Iga plokk peab olema kasutusel kas täpselt ühe faili poolt, või vabade plokkide listis
- Muu meta-info kooskõla kontrollimine
  - Näit. ebatavalised faili õigused nagu `007`

Taastumine vigadest

- Ketta kopeerimisel varundusseadmele kaks strateegiat:
  - ”Füüsiline kopeerimine” (*physical dump*) — lindile kirjutatakse üksteise järel kõik ketta plokid
    - Lihtne realiseerida ja kiire
    - Vigased plokid
    - Ebaefektiivne lindi kasutus — näit. ”varundatakse” ka vabad plokid
  - ”Loogiline kopeerimine” (*logical dump*) — alustades etteantud kataloogidest, rekursiivselt kopeerib varundamist vajavad failid ja kataloogid
    - Toetab inkrementaalset varundamist
    - Mitte-varundatavaid faile ei kopeerita

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 pidada nii metainfo kui kogu info kohta
- Logi võib asuda samal seadmel või hoopis teisel seadmel kiiruse huvides

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

NFS — Network File System

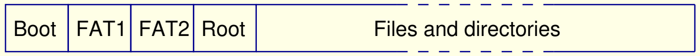
- Originaalselt Suni (ONC+) võrgufailisüsteem, praktikas *de facto* standard *nixite vahel failide jagamiseks
- SunRPC (ONC RPC) üle UDP (LAN puhul) või TCP (WAN puhul)
- Klient-server mudel
- Klient omab failisüsteemidraiverit, mis oskab suhelda serveritega
- Server ekspordib oma failisüsteemist mingeid 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 kasutavates masinates

NFS: protokollid

- Failide jagamise protokoll (NFS protokoll ise)
  - Saab RPC kaudu süsteemifunktsioonidele analoogseid protseduuriväljakutseid
  - Faile eristatakse failipidemetega
  - Olekuvaba (*stateless*) failiserver
  - Kataloogist saab alamkataloogide ja failide kohta failipidemeid edasisteks operatsioonideks
  - NFS v2 sünkroonne, v3 laseb eriliidese järgi kasutada ka asünkroonset liidest
- Monteerimise protokoll — annab esialgse failipideme jagataud kataloogi tipu kohta, kui on lubatud
- Muud protokollid — lukustamine, quota, . . .

Näide: MS-DOSi failisüsteem

- Kettaruumi kaldamiseks kasutatakse FAT-i
- Failisüsteemi paigutus kettal:



- Kolm versiooni — FAT-12, FAT-16, FAT-32
  - Ketta-aadressi pikkus vastavalt 12, 16 ja 32 bitti
  - FAT-32 korral tegelikult kasutusel 28 bitti
- Kettaploki suurus 512 B – 32 kB

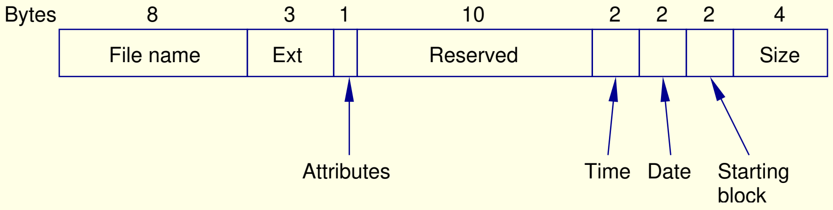
Näide: MS-DOSi failisüsteem

- Maksimaalne partitsiooni suurus:

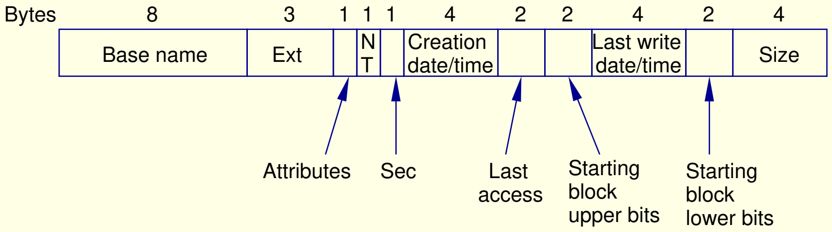
Ploki suurus	FAT-12	FAT-16	FAT-32
0.5 kB	2 MB		
1 kB	4 MB		
2 kB	8 MB	128 MB	
4 kB	16 MB	256 MB	1 TB
8 kB		512 MB	2 TB
16 kB		1024 MB	2 TB
32 kB		2048 MB	2 TB

Näide: MS-DOSi failisüsteem

- Kataloogikirje struktuur (FAT-12 ja FAT-16):

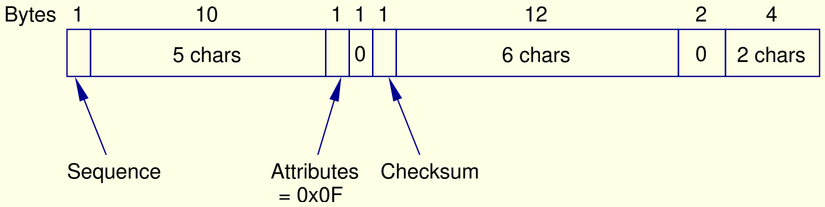


- Kataloogikirje struktuur FAT-32s (Windows 95/98/ME):

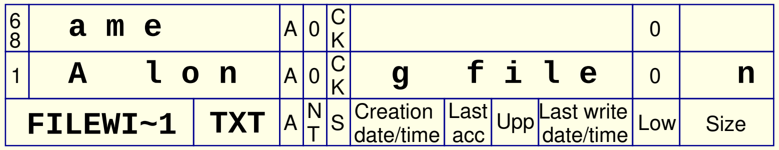


Näide: MS-DOSi failisüsteem

- Pikad failinimed FAT-32s:



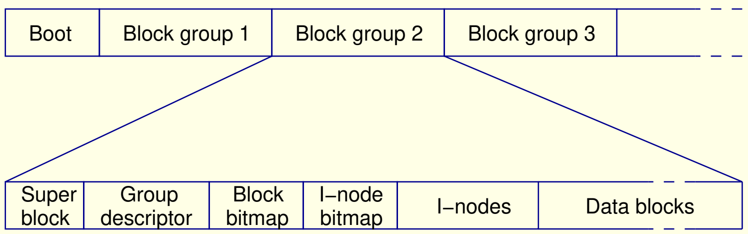
- Näide:



- Maksimaalne nime pikkus 260 märki

Näide: Ext2 failisüsteem

- Linuxi standartne failisüsteem
- Ketas (partitsioon) jaotatud plokirühmadeks
- Failisüsteemi paigutus kettal:



- Superplokk — plokkide ja i-kirjete arv, plokkide suurus, . . .
- 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 ext2 failisüsteemis on 128 baiti
- I-kirjes on meta-info objekti kohta (v.a. objekti nimi) ning viidad andmeplokkidele (12 otseviita ning 3 kaudset viita)
- Meta-info sisaldab:
  – Objekti tüüp (fail/kataloog/sümbolviit/. . .) ja õigused
  – I-kirjele viitavate linkide arv
  – Omanik ja rühm
  – Faili suurus baitides
  – Viimati kasutamise ning muutmise ajad
  – . . .

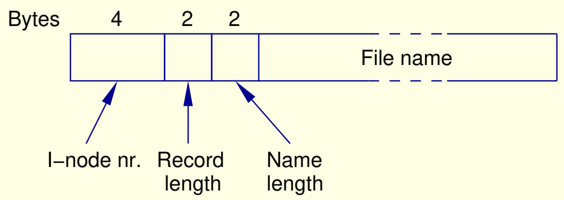
Näide: Ext2 failisüsteem

- I-kirjes on kokku 15 viita andmeplokkidele
  – 12 otseviita ning 3 kaudset viita (vastavalt ühe-, kahe- ja kolmetasemeliseks indekseerimiseks)
  – Kuni 60 baidise sümbolviida korral eraldi andmeplokki ei ole, vaid kasutatakse I-kirjes andmpeploki viitadele mõeldud välju
- Kõik I-kirjed tehakse failisüsteemi loomise ajal
  – Hiljem pole I-kirjete arvu enam võmalik muuta
- Ketta-aadress on 32 bitti; ploki suurus 1 – 8 kB

Ploki suurus:	1 kB	2 kB	4 kB	8 kB
Max. faili suurus:	16 GB	256 GB	2048 GB	2048GB
Max. f-süst. suurus:	2047 GB	8192 GB	16386 GB	32768 GB

Näide: Ext2 failisüsteem

- Kataloog on varieeruva pikkusega kirjete list
- Kataloogikirje struktuur:



- Uuemates versioonides lisaks veel objekti tüüp
- Maksimaalne nime pikkus 255 märki