Obsah

1. Název první kapitoly

1.1 Proč filesystémy

- jednotné API, odstínění od hardware
- odstínění od přistupování k sektorům přímo na disku či CD
- umožňuje přistupovat k různým hardwarům stejně

1.2 Jak fungují FS

- FS si rozdělí pamět na nejmenší možné jednotky (bloky-linux; alokační jednotky resp. clustry na windows)
- výhoda malých bloků–šetří místo,nejsou-li využity
- nevýhoda–defragmentace souborů

Definice (Soubor)**.** Persistentní úložiště dat. Má typické atributy(jméno(různá kódování-novější unicode(unix,ntfs))

Definice (Adresář)**.** Organiční jednotka, která obsahuje jiné adresáře, nebo soubory.

- Zvláštním druhem souboru jsou **sockety, roury** na unixu.
- Každý filesystem má kořenový adresář.
- Každý operační systém s filesystémem určuje pro každý proces aktuální adresář
- Záleží na filesystému, zda si aktuální adresář pro proces vytváří(fat32,ext=ano, ntfs=ne)

1.3 Operace na souboru

- čtení
- smazání(unlink-odebrání posledního hardlinku)
- seek (posun)

1.4 Mapování do paměti

- každý proces má svůj virtuální adresový prostor
- todo

1.5 Speciální soubory

- $IOCPL$ je operace, která je definovaná zvlášt na souborech jako cdromka etc
- na Windowsech $IOCPL$ na běžném souboru vypíše umístění k paměti

1.6 Linky

Definice (Hardlink)**.** Je odkaz do tabulky inodu(masterfiletable u ntfs)

Definice (Inode)**.** položka inode table(MFT u NTFS), která obsahuje vlastnosti: jméno, timestamp last modified, create, owner, práva,.. Obsahuje taky **odkaz.** na i**data** u soborů nebo na **seznam inodů** u adresářů

Uživatelské hardlinky si zakazují na adresáře kvůli následujícímu problému. Každý adresář a soubor musí být přístupný (přes jiné adresáře) z kořenového adresáře. Nechť povolíme hardlinky na adresáře. Mějme jediný hardlink e z nadřazeného adresáře na adresář D, který má podadresář SD. Tedy z D vede hardlink na SD.

Nechť uživatel přidá harlink z SD do D. Vznikla tedy kružnice mezi SD a D. Odebereme-li e neklesne počet hardlinků u D na nulu a tedy nebude smazán a přesto se na adresář D nelze dostat z kořenového adresáře.

Definice (Symlink)**.** Speciální soubor s cestou k jinému souboru, či adresáři. Filesystém vykoná všechny operace(kromě move a delete) místo na symlinku na souboru, na který symlink odkazuje. Symlink buď uvádí relativní, nebo absolutní cestu.

1.7 Ext2

- Jak vypadá jeden oddíl Ext2?
  1. bootovací část
  2. Group 0.
  3. ...
  4. Group n
- Group má velikost asi 32MB.
- Ext nebere jako nejmenší aloční jednotku sektor disku, ale **blok**.
- **Struktura groupy**
  1. Superblock-velikost bloku, groupy, info o disku-užitečný jen pro Groupu 0 za boot sektorem
  2. GDT-Group descriptor table
  3. I-tabulka inodů
  4. IB-bitmapa volných inodů
  5. B-bitmapa volných bloků
  6. Data

1.7.1 Jak se udržují data pro inode?

- Inode udržuje prvních 12 bloků v sobě
- 13, 14 blok jsou nepřímé-odkazují na bloky, které udržují odkazy na data
- 15 blok je nepřímý, nepřímý. Tj. Odkazuje na blok, který odkazuje na mezivrstvu bloků, z které se teprve dostaneme k blokům s adresama.
- $|inode| = min\{((\frac{b}{4})^2 + 2 * \frac{b}{4} + 12) * b; b * 2^{32}\}$
- Ext3 - má navíc žurnálování
- Ext4 - strom pro vyhledávání (todo-pro přístup k datům z inodu?)

1.8 FAT

- ansi jména souborů
- FAT jsou 3 filesystémy FAT32, 16, 12(diskety)
- todo proč zastaralý? jaké jsou limity

Struktura

- Reserved sectors
  - info o velikosti a počtu FAT, počtu cylindrů
  - obsahuje bootsector
  - u FAT 12, 16 má fat root directory, u FAT 32 je v boot sektoru napsané číslo kde se nachází root directory
- FAT(File Allocation Table)
  - většinou 2(kdyby se 1 rozbila)
  - 1 *zznamtabulky* odpovídá 1 *clusteru* v sekci DATA
  - vybrané hodnoty 1 záznamu:
    - 0 – volný blok
    - vadný blok
    - 1 – reservovaný blok
    - okolo MAXINT32 resp 16,12 – poslední blok v clusteru
- RD(root directory)
- DATA

1.8.1 Adresář

(fold)

- všechy informace o souboru 1 položka(krom rozšíření na dlouhá jména)
- obsahuje záznamy: jméno, datum, velikost ...
- jméno = 8 znaků + 3(přípona-todo(ano je příponam?))
- 1 cluster - todo(co s ním?)
- velikost souboru uložena v $4Bslov \rightarrow 4GBmaxvelikostsouboru$

1.9 NTFS

- spotřebuje 1kB na adresář
- úplně vše je soubor
- todo *bootsector* vs. *bootcluster*
- celý $NTFS$ na *clustery* – todo jinde to tak není?
- obsahuje $MasterFileTable$ v souboru MFT

Definice (Master File Table). • 1 záznam 1kb

- z *bootsektoru* se ví kde je MFT
- záznam s indexem 0 je samotná MFT
- z 0 se dozvím kde se nachází další(fragmentované) soubory MFT

Definice (Záznam MFT). • má pevnou délku

- struktura– $H - head$ a *seznamatribut*

Definice (Atribut záznamu MFT). • je objekt a obsahuje:

  - jméno
  - datum, čas
  - vlastník
  - oprávnění(staré NTFS)
  - velikost
  - obsahuje malou část dat

- dělení atributů zda se vejdou do záznamu:
  - *residentn* – celé v záznamu
  - *neresidentn* – odkazují ze záznamu na data, které definujíatribut

Definice (Adresář v NTFS)**.** Adresář je b^* neredundatní strom(nazývaný index) dle jména souborů. Kořen je v atributu záznamu–todo co to znamená, kde je tedy uložen kořenové struktury. Todo co obsahuje uzel? Todo co zde vlastně myslíme uzlem?

Definice (RUNLIST)**.** todo Co je runlist? zase se omlouvám Martinovi, ale nedokážu to definovat.

Když se nevejde *runlist* do *zznamu MFT* přidá se další atribut "rozcestník", který řekne kde jsou další části. *Bzovzznam* jsou všechny záznamy pro 1 soubor.

1.10 Obecné poznámy

Definice (řídký soubor)**.** todo–martin říkal, ale já to neumim zformulovat