Semestrální práce z předmětu KIV/ZOS

Max Nonfried
A19B0601P

11. února 2022

Obsah

Úvod

Zadáním této semestrální práce bylo vytvořit a naimplementovat zjednodušený souborový systém (dále jako "FS") na pricnipu i-uzlů. Podmínkou bylo, že program bude napsán v programovacím jazyce C (případně v C++).

Podrobný popis zadání a výpis jednotlivých funkcionalit lze nalézt v dokumentu *ZOS2021_SP.pdf*.

Pro implementaci jsem si vybral programovací jazyk C. Program je určen pro operační systém Windows.

Princip i-uzlů

I-uzel je datová struktura obsahující metadata o souborech a adresářích ve FS. V i-uzlu jsou uloženy informace jako například velikost souboru a v jakých datových blocích se nachází. Způsob uložení informace, v jakých datových blocích se soubor nachází, je zajištěno dvěma způsoby. Buď přes přímé odkazy na datové bloky, které jsou uloženy přímo v i-uzlu, nebo přes odkazy nepřímé (v i-uzlu je odkaz na blok, kde jsou uloženy odkazy na bloky). Princip odkazů je znázorněn na obrázku 3.2. Jméno souboru je pak uloženo jako dvojice "jméno - adresa i-uzlu"v adresáři. Uzel pro adresář a uzel pro soubor je stejný. Na principu i-uzlů jsou založeny souborové systémy `Extended filesystem`, které jsou používány v linuxových systémech.

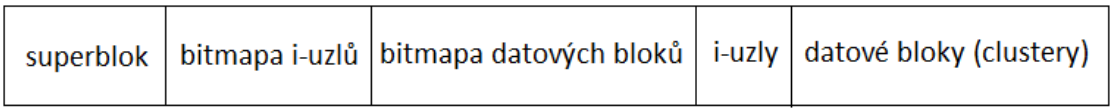
Struktura FS

Pořadí uložení jednotlivých částí FS je popsáno níže na obrázku 3.1.

Adresy začátků jednotlivých částí jsou uloženy v superbloku. Superblok je vytvořen při formátování. Po každém spuštění programu je načten do paměti (za předpokladu, že soubor s FS již existuje).

V bitmapách je uložena informace, zda je daný uzel či datový blok obsazen, nebo ne (obsazen: 1, volný: 0). Jsou naimplementovány tak, že jeden bit je uložen v jednom bytu. Velikost jedné bitmapy v bytech tedy odpovídá počtu i-uzlů.

Následují i-uzly a datové bloky. I-uzlů a datových bloků je stejný počet. I-uzel souboru a i-uzel adresáře lze rozeznat pomocí boolu `isDirectory` ve struktuře `inode`. Jedna položka adresáře je uložena v jednom datovém bloku. Datový blok má velikost 4096 B.



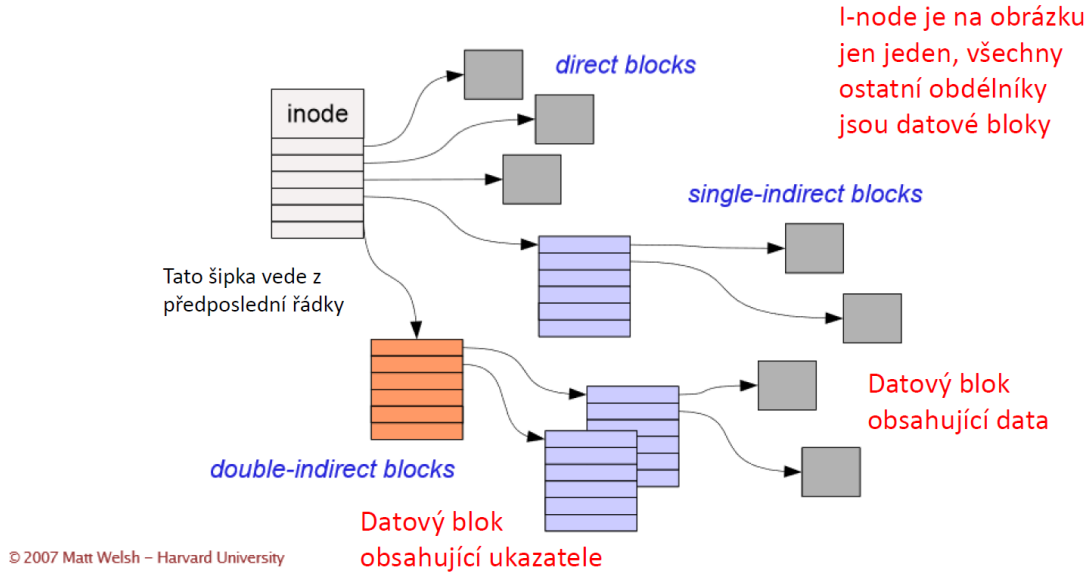
Obrázek 3.1: Nákres struktury uložení jednotlivých částí FS

Počet i-uzlů a tedy i datových bloků závisí na celkové velikosti FS. Vzorec pro výpočet je:

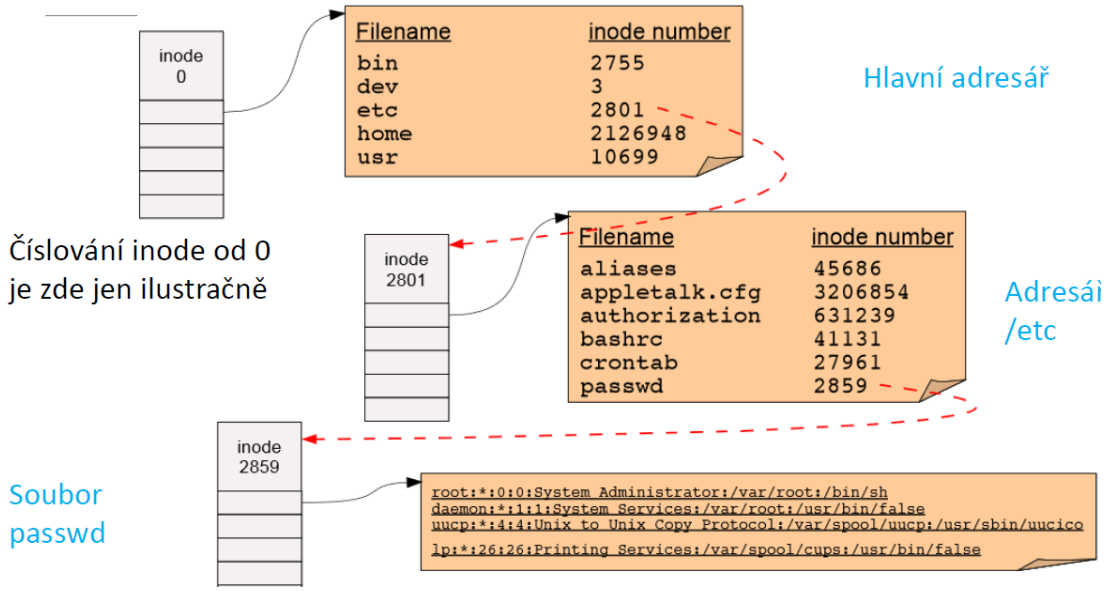
$$\frac{celkova\ velikost\ FS\ -\ velikost\ superbloku}{2\ +\ velikost\ iuzlu\ +\ velikost\ datoveho\ bloku}$$

3.1 Odkazy

Princip přímých a nepřímých odkazů je popsán níže na obrázcích 3.2 a 3.3. Přímé odkazy v sobě rovnou obsahují adresu bloku, kde jsou data souboru. Jednoduchý nepřímý odkaz v sobě obsahuje adresu bloku, kde jsou uloženy odkazy na bloky, kde jsou uloženy data. Dvojitý nepřímý odkaz obsahuje adresu na blok, kde jsou uloženy odkazy na bloky, ve kterých jsou uloženy data souboru.



Obrázek 3.2: Přímé a nepřímé odkazy, zdroj: 6



Obrázek 3.3: Vyhledávání pomocí i-uzů, zdroj: 6

Popis jednotlivých modulů

4.1 definitions.h

Hlavičkový soubor, který je importovaný do všech céčkových souborů. Obsahuje importované knihovny, definuje konstanty, deklaruje globální proměnné a importované hlavičkové soubory jednotlivých céčkových souborů.

4.1.1 Popis důležitých struktur

Struktura `superblock` obsahuje informace o FS (např. počet clusterů, velikost clusteru, počáteční adresy bitmap, uzlů, datových bloků). Tato struktura je nahrána z FS do paměti při spuštění programu. Struktura `inode` představuje jeden uzel. Obsahuje např. číslo uzlu, velikost uzlu, odkazy na datové bloky, atd. Struktura `directory_item` představuje položku adresáře. Tou může být jak soubor, tak jiný adresář.

4.2 main.c

Soubor obsahuje funkci `main(...)`, funkci pro přípravu programu (otevření FS, inicializace globálních proměnných) a funkci pro uklizení po skončení programu. Jestli příprava proběhne v pořádku, spustí se funkce `listening(...)`, ve které běží nekonečná smyčka pro přijímání příkazů.

4.3 input_handler.c

Soubor obsahuje funkci `listening(...)`, která zajišťuje přijímání příkazů z konsole. Po přijetí příkazu se funkcí `parse_input(...)` rozpozná o jaký příkaz se jedná. Podle toho se zavolá příslušná funkce pro jeho zpracování.

Nekonečný cyklus ve funkci `listening(...)` lze přerušit příkazem "exit".

4.4 commands_functions.c

Obsahuje funkce které zajišťují vykonání jednotlivých příkazů. Funkce jsou po příkazech pojmenovány. Volání těchto funkcí probíhá z funkce `parse_input(...)`,

která jim předá vstup od uživatele rozdělený mezerama.

4.5 get_functions.c

Soubor obsahuje všechny funkce, které zajišťují funkcionality typu `get`. Tedy např. `get_inode(...)`, `get_free_inode(...))`, `get_block_to_write(...)`.

4.6 set_functions.c

Soubor obsahuje všechny funkce, které zajišťují funkcionality typu `set`. Tedy např. `set_actual_path(...)`, `set_actual_inode(...)`, `set_bit_bitmap_data(...)`.

4.7 read_and_write_functions.c

Soubor obsahuje všechny funkce, které nějakým způsobem zajišťují čtení, nebo zápis. Tedy např. `write_directory_item(...)`, `read_file_from_fs(...)`, `write_item_name_to_console(...)`.

4.8 create_functions.c

Soubor obsahuje všechny funkce, které mají v názvu "create". Tedy např. `create_inode(...)`, `create_superblock(...)`, `create_bitmaps(...)`.

4.9 other_tools.c

Soubor obsahuje zbylé funkce, které neodpovídaly zařazení do ostatních souborů. Příkladem můžou být funcke pro přiřazování nových datových bloků do uzlů, funkce pro odstranění souboru či složky a různé pomocné funkce.

Spuštění aplikace

Program se spouští s jedním parametrem. Parametr je cesta k souboru, kde má být souborový systém. Program se překládá nástrojem GNU Make. Pro správný překlad je zapotřebí verze 4.2.1. Příkaz pro přeložení:

```
mingw32-make -f Makefile
```

Příkaz pro spuštění (parametry jsou uvedeny v <>):

```
./main <cesta k souboru, kde má být souborový systém>
```

Příkazy odpovídají příkazům uvedených v souboru se zadáním, tedy v *ZOS2021_SP.pdf*. Program se ukončuje příkazem:

```
exit
```

Závěr

Program se mi podařilo s drobnými nedostatky úspěšně naimplementovat. Během práce jsem se seznámil s principem fungování souborových systémů na principu i-uzlů a prohloubil své zkušenosti s programovacím jazykem C.

Možné vylepšení vidím například v rozšíření pro platformu Linux. To by zahrnovalo upravit kód, který načítá vstup od uživatele, kde je problém s textovým kódováním ukončovacích znaků.

Seznam obrázků

Literatura

1] *zos0708_2021.pdf*, Ladislav Pešička, přednáška z předmětu KIV/ZOS