Driver sistem de fisiere

Termen de predare: Duminică 24 Mai, ora 23.59 (2009-05-24)

Cuprins

Enun

Să se scrie un modul de kernel care să implementeze sistemul de fiiere `PITIX`. Acest sistem de fiiere va oferi suport doar pentru fiiere i directoare. Nu se vor implementa operaiile de suport pentru link-uri hard sau simbolice. De asemenea, nu se vor implementa operaiile de suport pentru fiiere speciale (pipe-uri, dispozitive caracter sau bloc). Practic va trebui să implementai următoarele operaii:

- pentru directoare: `lookup`, `unlink`, `mkdir`, `rmdir`, `readdir`
- pentru fiiere: `create`, `truncate`, funcia de bitmap (vezi `minix_get_block`)

Restul funciilor fie au implementări generice în kernel, fie nu trebuie să le implementai.

Structura pe disk a sistemului de fiiere este prezentată mai jos:

unde:

- `Superblock` conine superblocul (4096 octei)
- `Imap` conine harta de bii (bitmap) a blocurilor ocupate de inode-uri (1 bloc)
- `Dmap` conine harta de bii (bitmap) a blocurilor ocupate de date (1 bloc)
- `Izone` conine inode-urile (32 blocuri)
- `Dzone` conine datele (coninutul efectiv al fiierelor) (8*block_size blocuri)

Structura superblocului (pe disc) este descrisă de următoarea structură:

```
struct pitix_super_block {
        unsigned long magic;
        __u8 version;
        __u8 inode_data_blocks;
        __u8 block_size_bits;
```

```
        __u8 imap_block;
        __u8 dmap_block;
        __u8 izone_block;
        __u8 dzone_block;
        __u16 bfree;
        __u16 ffree;
//...
}
```

unde

- `magic` trebuie să fie iniializat cu `PITIX_MAGIC`
- `version` trebuie să fie iniializat cu 1
- `inode_data_blocks` este numărul de blocuri de date din inode (vezi mai jos la descrierea inode-ului)
- `block_size_bits` este dimensiunea blocului în puteri ale lui doi; dimensiunea blocului trebuie să fie 512, 1024, 2048, sau 4096.
- `imap_block` este numărul blocului (relativ la dispozitiv) pentru vectorul de bii folosit pentru alocarea/eliberarea inode-urilor
- `dmap_block` este numărul blocului (relativ la dispozitiv) pentru vectorul de bii folosit pentru alocarea/eliberarea blocurilor de date
- `izone_block` este numărul primului bloc (relativ la dispozitiv) al zonei de inode-uri
- `dzone_block` este numărul primului bloc (relativ la dispozitiv) al zonei de date
- `bfree` este numărul de blocuri libere (nealocate)
- `ffree` este numărul de inode-uri libere (nealocate)

Inode-urile vor fi stocate în zona de inode-uri i sunt descrise de următoarea structură:

```
struct pitix_inode {
        __u32 mode;
        __u32 uid;
        __u32 gid;
        __u32 size;
        __u32 time;
        __u16 data_blocks[0];
};
```

unde

- `mode` reprezintă drepturile de acces i tipul inode-ului (fiier sau director) aa cum sunt reprezentate în kernel
- `uid` reprezintă UID-ul aa cum este el reprezentat în kernel
- `gid` reprezintă GID-ul aa cum este el reprezentat în kernel
- `size` reprezintă dimensiunea fiierului/directorului
- `time` reprezintă timpul de modificare aa cum este el reprezentat în kernel
- `data_blocks` este un vector (cu dimensiunea precizată in superbloc) care indică numerele blocurilor de date relativ la zona de date; intrările nefolosite trebuie setate pe 0 (adică dacă avem un fiier cu dimensiune 2050 i dimensiunea blocului de 1024 atunci vom avea primele trei valori din vector strict pozitive i restul 0)

Directoarele au asociate un singur bloc de date în care vor fi stocate intrările de director. Acestea sunt descrise de următoarea structură:

```
struct pitix_dir_entry {
        __u32 ino;
        char name[PITIX_NAME_LEN];
};
```

unde

- `ino` reprezintă numărul inode-ului fiierului sau directorului; acest număr este un index în zona de inode-uri
- `name` reprezintă numele fiierului sau al directorului; lungimea maximă a numelui este 16 octei (`PITIX_NAME_LEN`); dacă lungimea numelui e mai mică decât 16 octei, atunci numele va fi terminat cu caracterul ASCII ce are codul 0 (la fel ca la irurile de caractere)

Toate valorile numerice sunt stocate pe disc în CPU byte-order.

Directorul rădăcină va avea alocat inode-ul 0 i blocul de date 0.

Pentru simplificare, la `mkdir` nu este necesară crearea intrărilor `"."` i `".."` în noul director; tester-ul se folosete de această presupunere.

Precizări Linux

- un exemplu de imagine, ce conine aceste fiiere
- structurile descrise le putei găsi i în acest header
- sursele sistemului de fiiere minix vă pot fi de folos

Testare

Pentru simplificarea procesului de corectare al temelor, dar i pentru a reduce greelile temelor trimise, corectarea temelor se va face automat cu ajutorul unor teste publice. Testele presupun că numele modulului de kernel este `pitix`.

Depunctări

În corectarea temei se va ine cont de următoarele criterii de depunctare:

- -1.0 warning-uri la compilare
- -1.0 implementare incorectă a cerinelor temei
- -0.2 lipsă `README` sau `README` necorespunzător
- -0.2 memory leaks
- -0.2 omiterea eliberării resurselor
- -0.1 lipsă comentarii sau comentarii nerelevante
- -0.1 cod neîngrijit
- -0.2 alte probleme constatate
- -0.0 observaii

În cazuri excepionale (tema trece testele prin nerespectarea cerinelor) i în cazul în care tema nu trece toate testele se poate scădea mai mult decât este menionat mai sus.

Notare

Datorită dificultăii ridicate, tema valorează 20 de puncte.

Resurse utile

1. Laboratorul 9. Drivere sisteme de fiiere (Linux) partea 1
2. Laboratorul 10. Drivere sisteme de fiiere (Linux) partea 2
3. Sursele sistemului de fiiere minix

Întrebări

Pentru întrebări legate de temă putei căuta, consulta sau trimite un e-mail pe lista de discuii (trebuie sa fii înregistrai).