PB152cv/07 Cvičenie 3

(Mapovanie do pamäte, mmap(), blokujúce vs. neblokujúce operácie)

Jozef Sabo

13. 10. 2025

Masarykova Univerzita, Brno

Cvičenie 03

Mapovanie do pamäte

zjednodušenie úpravy súboru

mapovanie súboru do pamäte

```
void *mmap(
    void *addr, size_t len, int prot, int flags,
    int fildes, off_t off
);
```

odovzdanie mapovaného zdroja

```
int munmap(void *addr, size_t len);
```

synchronizácia dát

```
int msync(void *addr, size_t len, int flags);
```

Mapovanie do pamäte

zjednodušenie úpravy súboru

mapovanie súboru do pamäte

```
void *mmap(
    void *addr, size_t len, int prot, int flags,
    int fildes, off_t off
);
```

odovzdanie mapovaného zdroja

```
int munmap(void *addr, size_t len);
```

synchronizácia dát

```
int msync(void *addr, size_t len, int flags);
```

Mapovanie do pamäte

```
void *mmap(
    void *addr, size_t len, int prot, int flags, int fildes, off_t off);
```

`void *addr` – adresa, kam odporúčame jadru uložiť mapovanie, používame `NULL`

`size_t len` – akú veľkosť chceme mapovať, zvyčajne veľkosť súboru

`int prot` – ochrana pamäte, zvyčajne `PROT_READ` alebo `PROT_WRITE` alebo kombinácia

`int flags` – určujeme ním typ mapovania

- `MAP_SHARED` – zmeny sa prejavia v mapovanom súbore
- `MAP_PRIVATE` – zmeny ostávajú viditeľné pre konkrétny proces

`int fildes` – popisovač otvoreného súboru, ktorý chceme namapovať

`off_t off` – pozícia, od ktorej chceme súbor mapovať (0 ak od začiatku)

Mapovanie do pamäte

```
void *mmap(
    void *addr, size_t len, int prot, int flags, int fildes, off_t off);
```

návratová hodnota je typu smerník, ten môže byť

- neplatný – hodnota -1 je pretypovaná na typ smerník, náhrada makro `MAP_FAILED`
- platný – od tejto adresy môžeme začať súbor používať ako bežnú pamäť

Mapovanie do pamäte

zjednodušenie úpravy súboru

mapovanie súboru do pamäte

```
void *mmap(
    void *addr, size_t len, int prot, int flags,
    int fildes, off_t off
);
```

odovzdanie mapovaného zdroja

```
int munmap(void *addr, size_t len);
```

synchronizácia dát

```
int msync(void *addr, size_t len, int flags);
```

Mapovanie do pamäte

zjednodušenie úpravy súboru

mapovanie súboru do pamäte

```
void *mmap(
    void *addr, size_t len, int prot, int flags,
    int fildes, off_t off
);
```

odovzdanie mapovaného zdroja

```
int munmap(void *addr, size_t len);
```

synchronizácia dát

```
int msync(void *addr, size_t len, int flags);
```

Mapovanie do pamäte

```
int munmap(void *addr, size_t len);
```

`void *addr` – adresa, ktorú vrátilo volanie `mmap()`

`size_t len` – veľkosť, ktorú chceme „odmapovať“, zvyčajne veľkosť súboru

ak vložíme do `size_t len` nesprávnu hodnotu, môže dôjsť k úniku pamäte

po volaní `munmap()` sa stáva pamäť neplatnou, zapis/čítanie vyvolá chybu segmentácie (SIGSEGV), podľa nastavenia hodnoty `flags` pri volaní `mmap()` sa vykoná synchronizácia s úložiskom

Mapovanie do pamäte

zjednodušenie úpravy súboru

mapovanie súboru do pamäte

```
void *mmap(
    void *addr, size_t len, int prot, int flags,
    int fildes, off_t off
);
```

odovzdanie mapovaného zdroja

```
int munmap(void *addr, size_t len);
```

synchronizácia dát

```
int msync(void *addr, size_t len, int flags);
```

Mapovanie do pamäte

zjednodušenie úpravy súboru

mapovanie súboru do pamäte

```
void *mmap(
    void *addr, size_t len, int prot, int flags,
    int fildes, off_t off
);
```

odovzdanie mapovaného zdroja

```
int munmap(void *addr, size_t len);
```

synchronizácia dát

```
int msync(void *addr, size_t len, int flags);
```

Mapovanie do pamäte

```
int msync(void *addr, size_t len, int flags);
```

`void *addr` – adresa, ktorú vrátilo volanie `mmap()`

`size_t len` – veľkosť, ktorú chceme synchronizovať

`int flags` – upravujú, ako sa zapíše súbor

- `MS_ASYNC` – neblokujúce, zapíše dáta asynchrónne
- `MS_SYNC` – blokuje kým sa zapíšu všetky dáta
- `MS_INVALIDATE` – zneplatní cache ostatných aplikácii s otvoreným súborom

Blokujúce a neblokujúce operácie

blokujúce

operácia sa zavolá a vykoná sa aspoň minimálna časť volania (napr. zapíše 1 bajt)

môže spôsobiť uviaznutie (smrtizámok, deadlock)

neblokujúce

operácia sa zavolá a ihneď navráti, oznámi aké množstvo práce vykonala

vykonať sa môže aspoň nejaká časť alebo nič

Neblokujúce operácie

vykonávajú sa zvyčajne na popisovačoch s príznakom `O_NONBLOCK`

návratová hodnota je s bežným významom okrem hodnoty `-1`, kde `errno` môže byť nastavené na `EAGAIN`, vtedy nejde o chybu, ale o zaneprázdnenosť

môžeme chcieť viaceré neblokujúce popisovače synchronizovať a/alebo blokovať

to docielime rozumným použitím volania `poll()`

`poll()` má v tomto momente zmysel aj na iba jeden popisovač