Cvičení ve 4. týdnu

Speciální soubory, vytváření, připojování a kontrola integrity souborového systému.

Speciální soubory

Speciální soubory (v UNIXovém slova smyslu) jsou objekty v systému souborů, které nejsou ani adresářem ani souborem obsahujícím data. Na disku nezabírají žádné datové bloky, jen vždy jeden i-node. Při čtení/zápisu dat z/do těchto speciálních souborů jsou jednotlivé požadavky na čtení/zápis dat delegovány příslušnému ovladači v jádře, popř. jinému spuštěnému procesu.

OS UNIX rozeznává tyto typy objektů (položek) v systému souborů:

znak výpisu ls	název	účel
-	file (soubor)	Klasický soubor - jediný objekt, který skutečně uchovává data
d	directory (adresář)	Podadresář aktuálního adresáře. Obsahuje další (vnořené) objekty, umožňuje vytvořit stromovou strukturu systému souborů.
l	link (symbolický link)	Odkaz na jiný objekt v systému souborů, zástupce
b	block device (blokové zařízení)	Read/Write operace s blokovým či znakovým zařízením jsou delegovány určitému ovladači v jádře, který s daty naloží tak, jak uzná za vhodné. Každé blokové či znakové zařízení obsahuje čísla MAJOR a MINOR (hlavní a vedlejší), které určují příslušný ovladač jádra. Blokové zařízení se od znakového liší tím, že používá buffer (cache) pro data.
c	character device (znakové zařízení)	
p	pipe (pojmenovaná roura)	Slouží k předávání dat mezi dvěma běžícími procesy. Funguje jako roura obyčejná s tím rozdílem, že ji smí otevřít i proces, který ji nezaložil.
s	socket (soket)	Síťové spojení procesů v adresovém prostoru AF_UNIX, komunikace je jen lokální v rámci jednoho počítače. Podobné pojmenované rouře. Socket umí zjistit a ověřit PID, UID a GID protějšího procesu a předat otevřené file descriptory.

Všechny tyto typy objektů (položek adresáře) systému souborů mají některé shodné vlastnosti. Tím je dána možnost použít pro práci s nimi stejná nebo podobná systémová volání, konstanty a metody práce. Společné vlastnosti jsou uloženy v i-node příslušného objektu:

- každý objekt má své jednoznačné jméno v systému souborů, lze ho tak identifikovat a odkazovat
- každý objekt má vlastníka a skupinu (používá se k omezení přístupu)
- každý objekt má stanovena přístupová práva (rwx) zvlášť pro vlastníka, skupinu a ostatní
- každý objekt nese jistá časová razítka (čas a datum vytvoření, modifikace, posledního přístupu, atd.)
- každý objekt může mít více jmen - tzv. hard linků. To znamená, že na týž i-node se může odkazovat více položek adresáře či adresářů.

UNIX používá konvenci pro pojmenování blokových a znakových zařízení. Jména však pro funkci nejsou určující, klíčem jsou MAJOR a MINOR čísla. Obě čísla se zobrazují při výpisu **ls -l /dev/***.

Soubor	MAJOR	MINOR	Význam
Příklady konvence pojmenování **blokových** zařízení			
/dev/hda	3	0	První pevný disk - IDE primary master. Přístup sektor po sektoru.
/dev/hda1	3	1	První partition prvního IDE disku

/dev/hdb	3	64	Druhý pevný disk - IDE primary slave.
/dev/fd0	2	0	První disketová mechanika
/dev/ram0	1	0	První ramdisk
/dev/sda	8	0	První SCSI disk
/dev/sda1	8	1	První partition prvního SCSI disku
Příklady konvence pojmenování **znakových** zařízení			
/dev/null	1	3	Zařízení zapomenutí. Co sem zapíšete, bude zapomenuto. Při čtení se chová jako soubor nulové délky.
/dev/audio	14	4	Zvukový kanál
/dev/kmem	1	2	Obraz kompletní paměti jádra
/dev/psaux	10	1	Vstupní konektor PS/2 myši.
/dev/tty1	4	1	První virtuální textová konsole
/dev/tty2	4	2	Druhá virtuální textová konsole
/dev/ttyS0	4	64	První sériová linka - COM1
/dev/zero	1	5	Zařízení binárních nul. Při čtení produkuje neustále znaky binární 0. Tváří se jako nekonečně dlouhý soubor obsahující jen binární 0.

Vytváření speciálních souborů
Bloková a znaková zařízení a pojmenované roury je možné vytvářet příkazem **mknod**. Před vytvořením blokového či znakového zařízení je nutné znát příslušná MAJOR a MINOR čísla příslušného ovladače v jádře. Někteří výrobci UNIXu tato čísla příliš nepublikují, pro Linux naleznete kompletní seznam v distribučním balíku každého jádra v souboru **/usr/src/linux/Documentation/devices.txt**.

Úkol 1: Pracujte na virtuálním počítači RaidFC4. Prostudujte syntaxi příkazu `mknod` a vytvořte nové blokové zařízení `/tmp/blabla` s MAJOR číslem 3 a MINOR číslem 65. Pak proveďte následující příkazy:

```
mkdir /data
mount /dev/hdb1 /data
cp /etc/passwd /data
ls /data
umount /data

mkdir /test
mount /tmp/blabla /test
ls /test
```

Jak se dostal soubor do adresáře `/test`?

Systémy souborů
Každý disk, na který se ukládají soubory, obsahuje systém souborů (filesystem,FS). Struktura dat na disku, v jaké jsou soubory ukládány se však liší pro každý OS. Každá tato metoda (typ FS) má své jméno:

Operační systém	Jméno FS	Pozn.
UNIX (obecně)	ufs	Prvotní ufs zavedl pojmy i-node a datový blok. Dnes se používají různé vylepšené klony ufs.

Minix	minix	Jednoduchý FS, omezení délky jmen (16zn), velikost disku do 64MB, dnes vhodný pro diskety či ramdisk
Linux	ext2	Propracovaný FS odvozený od minix, prakticky bez omezení délky jména (256zn), soubory do 4GB, disk do 17TB.
MS-DOS	dos	Velmi omezený FS - jména 8zn+3zn přípona, disk do 8GB, jen 32-64k alokačních jednotek - velké plýtvání místem
Windows 95	vfat	Rozšíření dos FS o dlouhá jména, zpětně kompatabilní
	fat32	Rozšíření vfat. Vnitřní struktury jsou nyní 32-bitové. Velké zvýšení hranic velikosti souboru a disku, snížení plýtvání místem při alokaci, nevýhody systému FAT zůstávají
Windows NT	ntfs	32-bitový FS pro Windows NT odvozený z hpfs.
CD-ROM	iso9660	Takto jsou uložena data na všech CD-ROM
OS/2	hpfs	32-bitový FS vyvinutý pro OS/2

Většina OS podporuje jen svůj FS a iso9660 někdy ještě jeden či dva jiné. OS Linux je tak trochu vyjímkou, podporuje velké množství různých FS.

Krom těchto skutečných FS existují ještě **síťové FS**, které nikam nic neukládají, ale transportují požadavky po síti. Typickým příkladem je NFS známý z UNIXového světa, NCP v prostředí DOS/ NOVEL nebo SMB (samba) pro MS Windows, ale i méně známá CODA pro zabezpečenou komunikaci. Síťové FS však neorganizují data do bloků disku, proto na fileserveru musí existovat vždy ještě skutečný FS, pod kterým se data ukládají.

Další skupinou jsou tzv. **virtuální FS**, které neslouží k ukládání souborů. Příkladem je PROC FS, který zobrazuje informace z jádra OS a umožňuje některé parametry měnit zápisem do příslušných souborů. Soubory v proc-FS neexistují, jádro je vždy vygeneruje dynamicky pro každý přístup. Dalším příkladem je pts FS nebo dev FS, který obsahuje jen speciální zařízení, připojuje se do adresáře /dev a umožňuje jádru aktuálně měnit seznam spec. souborů v /dev. OS Solaris používá tmp FS pro adresář /tmp. Je implementován tak, že využívá k uložení primárně volnou paměť a tím zrychluje přístup k dočasným souborům v /tmp.

Spolupráce FS
OS UNIX nepoužívá zvláštní adresářový strom pro každý používaný disk (jako např. DOS nebo Windows používají označení disku A: B: C: atd.), ale všechny používané FS spojuje do jediného stromu. Při takzvaném připojení disku (FS) do nějakého adresáře se namapuje celý adresářový strom dotyčného disku jako podstrom daného adresáře.

Příkaz **mount** bez parametru vypíše seznam aktuálně připojených FS. Vypište aktuální stav na vašem počítači.

Vytvoření FS se provádí příkazem **mkfs** (newfs u Solaris a HPUX). Příkaz mkfs má parametr -t specifikující typ FS a další argument speciální zařízení určující hostitelský disk (resp. partition). Vytvoření FS se někdy zaměňuje s pojmem formátování disku. Formátování je proces vyznačení jednotlivých sektorů na disku či disketě, je nutné zapsat celý disk, ale po naformátování je disk stále nečitelný, protože neobsahuje FS. Vytvoření FS znamená inicializovat na disku struktury FS - tabulky popisující soubory v kořenovém adresáři a volné bloky - tedy jen několik málo sektorů. Formátování je činnost zdlouhavá, kterou však moderní pevné disky nevyžadují a diskety již jsou většinou naformátované. Vytvořit FS je nutné vždy po přerozdělení disku na jednotlivé partition (logické disky).

Úkol 2: Vytvořte nový FS: Jako místo uložení použijte partition `/dev/hdd1`. Např. ext2 vytvoříte buď
```
mkfs -t ext2 /dev/hdd1
```
nebo
```
mke2fs /dev/hdd1
```

Program mkfs je jen nadstavba nad programy mke2fs, mkdosfs, mkfs.minix, ..., které vytvářejí příslušné FS.

Připojení FS se provádí příkazem
`mount -t typ zařízení adresář`
Parametr `-t` určuje typ FS. Většina FS je automaticky detekována, takže není nutné ho používat. Zařízení je blokové zařízení identifikující nosič dat (partition disku, disketa, cdrom ...), adresář určuje místo, kam se napojí příslušný adresářový strom.

Úkol 3: Vytvořte adresář `/pokus`. Připojte vámi vytvořený FS do adresáře `/pokus`. Opětné odpojení provedete příkazem
`umount /pokus`
Ověřte střídavým připojováním a odpojováním a vytvářením souborů v `/pokus`, zda jsou soubory skutečně ukládány na ramdisku.

POZOR! Odpojit lze pouze FS, který není zrovna používán (!) to znamená, že žádný soubor v něm nesmí být otevřen, žádný program zněho není spuštěn a žádný program nemá žádný jeho podadresář nastaven jako aktuální. Proto se NIKDY NEPODAŘÍ odpojit kořenový FS. Ani `/usr` se vám nepodaří odpojit dokud nepozabíjíte všechny procesy, které ho používají. PID procesů, které brání odpojení FS zjistíte příkazem `fuser -m`.

Trvalé připojení, tj. taková změna konfigurace, kdy se váš nový FS připojuje automaticky vždy po rebootu, zajistíte přidáním příslušné řádky do souboru `/etc/fstab`. Každý řádek obsahuje zařízení, adresář k připojení, typ FS, atributy připojení, příznak dump a pořadí provádění automatické kontroly. Detaily viz `man fstab`.

Úkol 4: Upravte `/etc/fstab` tak, aby se adresář `/pokus` připojoval automaticky. Ověřte správnost restartem virtuálního počítače.

Kontrola FS se provádí příkazem `fsck`. Parametrem je blokové zařízení. Program fsck je opět nadstavba nad programy e2fsck, dosfsck, atd. Kontrolovat lze jen FS, které zrovna nejsou připojeny. V případě nouze lze kontrolovat FS připojené s atributem read-only, což nám pomůže např. při kontrole kořenového (root) FS, pokud nelze nastartovat OS z jiného zdroje.

Úkol 5: Zkontrolujte nyní FS na vašem ramdisku. (nápověda: odpojte FS pomocí `umount` a zkontrolujte pomocí `fsck` nebo `e2fsck`)

Program fsck vypíše hlášení, že FS je clean, ale kontrolu neprovede (FS byl správně odpojen a je tedy konzistentní). Skutečnou kontrolu `fsck` provede, jen pokud počet připojení/odpojení FS od poslední kontroly překročil určitou mez, nebo uběhla stanovená maximální doba od poslední kontroly, nebo kontrolu explicitně vyžádáme přepínačem -f.

Úkol 6: Proveďte skutečnou kontrolu příkazem: `fsck -f /dev/ram0`

Bodovaný úkol 7: Zkontrolujte kořenový FS.
Postup:
Kořenový FS nelze odpojit, restartujeme proto virtuální počítač do single-user módu a změníme atribut jeho připojení na read-only (zadáním parametru jádra S v zavaděči GRUB). Změna připojení jen pro čtení se provede příkazem:
`mount -o remount -r -n /`
Parametr **-o remount** znamená, že FS je již připojen, požadujeme změnu. Přepínač **-r** znamená read-only, přepínač -n říká, aby program mount nezapisoval aktuální stav připojení do souboru /etc/mtab (protože FS již v tu dobu bude read-only a zápis by se nepodařil). Skutečnost, zda je FS read-only, můžeme ověřit vytvořením souboru nebo adresáře, což by se nemělo podařit ani uživateli root. Nyní lze zkontrolovat popř. opravit kořenový FS programem `fsck`. Pokud se vyskytly chyby, je v případě kořenového FS vhodné restartovat po opravě počítač. Pokud žádné chyby nebyly, je na čase změnit připojení na read-write
`mount -o remount -w -n /`
a po případné kontrole ostatních FS přepnout do normálního runlevelu.