# Co umí souborové systémy

**Jan Kasprzak**

E-mail: kas@fi.muni.cz

**Klov slova:** Linux, storage, file system

**Abstrakt**

*Souborový systém je jednou ze základních komponent UNIXového počítače. V této přednášce shrneme historii souborových systémů v UNIXu i Linuxu, představíme souborové systémy, které se reálně používají v současnosti v Linuxu (ext3, XFS, JFS, ReiserFS, JFFS2, OCFS2, GFS2), popíšeme pokročilé a nestandardní vlastnosti některých souborových systémů, a ukážeme, kam směřuje vývoj, včetně představení projektů, které se vize budoucnosti snaží realizovat: ext4, Reiser4, CRFS/BTRFS, POHMELFS, UBIFS a některé další.*

**Abstract**

*The File system is one of the principal components of the UNIX computer. In this paper, we will summarize a history of the file systems in both UNIX and Linux, we will introduce the file systems currently in use in Linux (such as ext3, XFS, JFS, ReiserFS, JFFS2, OCFS2, GFS2), we will describe advanced features of some file systems, and we will show possible directions of future development in this are: ext4, Reiser4, CRFS/BTRFS, POHMELFS, UBIFS, and some others.*

# 1 Základní služby souborových systémů

Operační systém UNIX, stejně tak jako jeho ideový následník Linux, používá k ukládání dat na diskové jednotky datovou strukturu, nazývanou *souborový systém* (file system, FS). Úlohou souborového systému je zavést na blokovém zařízení jako je pevný disk, disketa, či dnes stále více rozšířená flash paměť strukturu, která umožní využívat blokové zařízení jako úložiště stromu souborů a adresářů.

Takovéto struktuře říkáme také *metadata* (na rozdíl od *dat*, což už jsou uživatelem uložené informace uvnitř souborů). Jednotlivé souborové systémy se pak liší zejména tím, jak vypadají jimi používaná metadata, ale také tím, jakým způsobem s metadaty pracují.

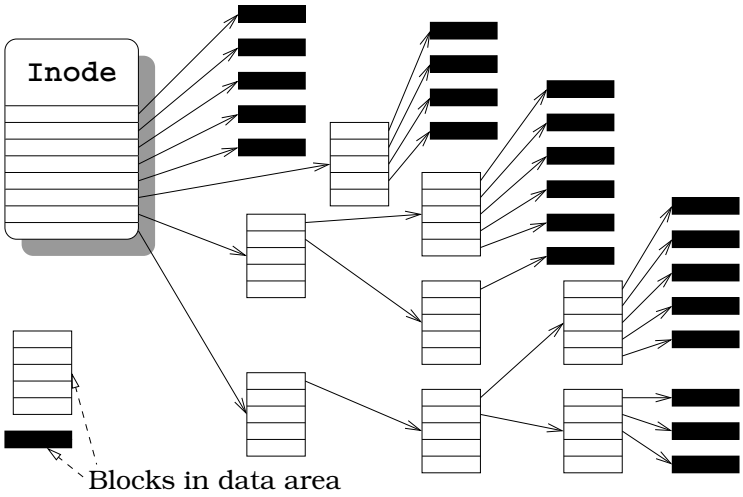
Souborové systémy v UNIXu jsou ukládány na nezávislých blokových zařízeních (blokové zařízení je typicky oblast – *partition* – pevného disku) tak, že každé blokové zařízení se souborovým systémem na něm tvoří samostatný strom souborů a adresářů, tak zvaný *svazek* (volume). Jednotlivé svazky jsou pak spojeny do jednoho stromu souborů a adresářů, který je zpřístupněn procesům, běžícím na UNIXovém počítači.

## 1.1 I-uzly

Ve všech UNIXových souborových systémech se v různých podobách vyskytuje datová struktura nazývaná *i-uzel* (identifikační uzel, i-node). Tato struktura obsahuje metadata o jednom konkrétním souboru a je tedy jakousi „hlavičkou“ souboru. I-uzel je identifikován svým číslem, které je v rámci jednoho svazku jednoznačné. V i-uzlu jsou uložena přístupová práva souboru, vlastník a skupina souboru, typ souboru (běžný soubor, adresář, roura a další), časy (modifikace souboru, přístupu k souboru a modifikace i-uzlu), počet odkazů, délka souboru, a také odkaz na datové bloky souboru.

Datové bloky bývají z i-uzlu odkazovány různě. Tradiční UNIXové souborové systémy používají pseudologaritmickou strukturu naznačenou na obrázku 1: přímo v i-uzlu je třináct ukazatelů na datový blok. Prvních deset ukazatelů ukazuje přímo na prvních deset bloků souboru. Jedenáctý ukazatel je tzv. první nepřímý odkaz – odkaz na blok v datové oblasti, který je celý rozdělen na ukazatele na skutečné datové bloky souboru. Dvanáctý ukazatel funguje podobně, jen bloky jsou odkazovány přes dvě úrovně. Třináctý ukazatel je pak třetí nepřímý odkaz.[1]

[1] Máme-li například bloky velikosti 1 KB a 32-bitové (čtyřbajtové) ukazatele na blok, pak v i-uzlu je prvních 10 bloků (10 KB) odkazováno přímo, dalších 1 KB/4 B = 256 bloků odkazováno přes první nepřímý ukazatel, $256^2$ bloků (tedy 64 MB) přes druhý nepřímý ukazatel a $256^3$, čili 16 GB přes třetí nepřímý ukazatel. Maximální velikost souboru v takovémto souborovém systému je tedy o něco více než 16 GB.




Obrázek 1 I-uzel standardního UNIXového souborového systému

Tato struktura má několik zajímavých vlastností:

- Krátké soubory a začátky dlouhých souborů maji menší režii a přístup k nim je rychlejší.
- Na druhé straně adresa kteréhokoli bloku souboru je zjistitelná pomocí nejvýše tří přístupů na disk.
- K přístupu ke kterémukoli datovému bloku souboru není třeba nejprve načítat předchozí bloky souboru (lze tedy efektivně provádět i nesekvenční, náhodný přístup k souboru).
- Je-li v i-uzlu odkazovaný nějaký datový blok, neznamená to, že nutně musí být odkazovány všechny předchozí bloky. V tom případě mluvíme o *souboru s dírou* (sparse file). Takovéto soubory mohou za jistých okolností šetřit místo na disku. Díra se při čtení tváří jako bloky nulových bajtů, při zápisu se teprve alokuje místo na disku.[2]

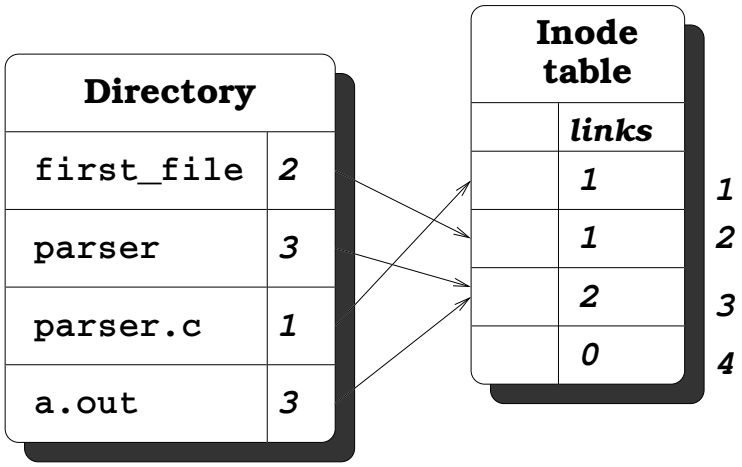
[2]Příkladem souboru s dírou může být soubor `lastlog` (obvykle `/var/log/lastlog`). Vyzkoušejte na něm příkaz `ls -ls` – tento příkaz vypíše počet skutečně alokovaných bloků souboru i velikost souboru (offset posledního platného bajtu).

### 1.2 Adresáře

Adresář (*directory*) je v UNIXu v podstatě běžný soubor, jen má u sebe příznak, že jde o adresář. Obsah tohoto souboru je interpretován jako seznam dvojic: pro každý soubor je zde jméno souboru a číslo i-uzlu, který je pod tímto jménem dostupný. Takto zejména může být jeden soubor (i-uzel) dostupný pod více jmény, případně z více adresářů na tomtéž svazku (tzv. *pevný odkaz*, hard link).

Každý adresář obsahuje položku „`.`“ (odkaz sám na sebe) a položku „`..`“ – odkaz na nadřazený adresář (v kořeni svazku odkaz sám na sebe).

Jméno souboru se v UNIXu interpretuje s rozlišováním velikosti písmen a může obsahovat libovolné znaky kromě lomítka (používá se pro oddělení komponent jména v rámci cesty) a nulového bajtu (používá se pro ukončení řetězců v jazyce C). Starší souborové systémy povolovaly jména souborů do délky 14 znaků, dnešní mají limit někde okolo 256 znaků. UNIX neeviduje kódování (znakovou sadu) jmen souborů, na dnešních systémech se obvykle používá UTF-8 pro svoji kompatibilitu s ASCII i univerzálnost.

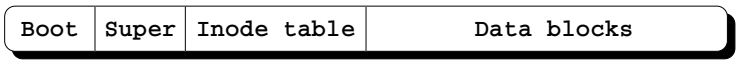



Obrázek 2 Struktura adresáře

## 2 Historie

### 2.1 System V File System

Souborový systém z UNIXu System V (`s5fs`) může posloužit jako příklad toho, jaké minimální vlastnosti dnešní UNIXový souborový systém potřebuje.




Obrázek 3 Struktura souborového systému `s5fs`

S5FS nechával na začátku volný blok (tzv. *boot block*) pro zavaděč systému. Dále následoval *superblok* se sumárními informacemi o tomto souborovém systému. Pak tabulka i-uzlů a nakonec datové bloky. Vadné bloky souboru byly znepřístupněny tak, že byly alokovány pro některý vyhrazený i-uzel, který nebyl odkazován z žádného adresáře. Volné datové bloky byly ukládány jako zřetězený seznam, což po delším používání souborového systému vedlo k fragmentaci, a tím ke sníženému výkonu. Typická velikost bloku S5FS je 512 bajtů nebo 1 KB.

## 2.2 FAT

Souborový systém FAT (*File Allocation Table*) vlastně nemá s UNIXem ani Linuxem nic společného, i když Linux i většina dalších systémů s ním umí pracovat. Uvádíme jej jako ilustraci jiného než UNIXového přístupu k souborovým systémům.


Obrázek 4 Struktura souborového systému FAT

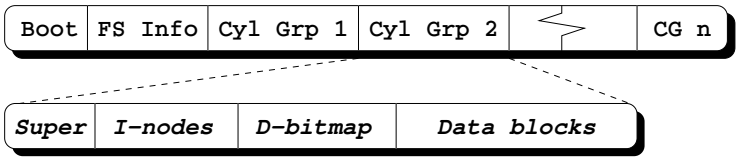
Souborový systém FAT začínal zaváděcím sektorem, dále byly dvě kopie alokační tabulky souborů a pak místo na datové bloky. V prvním datovém bloku začínal kořenový adresář. Alokační tabulka byla ve dvou kopiích z důvodu vyšší tolerance k výpadku. Alokační tabulka obsahovala jeden záznam (12-, 16- nebo 32-bitový) pro každý blok souborového systému (*cluster* v terminologii FAT). Tento záznam říkal, jestli je daný blok volný, vadný, nebo alokovaný v souboru nebo adresáři. V posledním jmenovaném případě pak záznam obsahoval číslo bloku, který po tomto bloku v daném souboru následuje (nebo informaci, že tento blok je posledním blokem daného souboru).

Adresář ve FAT obsahoval záznamy pevné délky (16 bajtů) se všemi potřebnými metadaty o souborech: 11 bajtů jména souboru interpretovaných jako osm znaků jména a tři znaky přípony bez ohledu na velikost písmen, čas modifikace souboru s přesností na dvě sekundy, odkaz na první datový blok souboru (další bloky se vyhledávaly pomocí alokační tabulky), atributy souboru (soubor nebo adresář, skrytý soubor, soubor jen pro čtení, atd.).

Pro kódování jmen souborů se v našich krajích používala kódová stránka IBM CP852 (i v systému Windows, který jinak data v souborech ukládal ve znakové sadě Windows-1250).

## 2.3 UFS

Souborový systém UFS (na některých systémech též FFS nebo EFS) pochází z BSD UNIXu. Autoři se snažili odstranit některé nedostatky S5FS. Celý souborový systém je rozdělen na několik úseků – *cylinder group*, CG (podle geometrie disku) – které obsahují kopii superbloku pro případ poškození primárního superbloku, část tabulky i-uzlů, nově bitmapu volných datových bloků a část datových bloků samotných.




Obrázek 5 Struktura souborového systému UFS

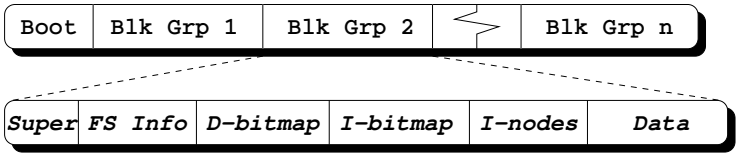
Alokační strategií je zajištěno, že v případě volného místa nevzniká fragmentace ani v případě, že se paralelně vytváří více souborů (každý je pak vytvářen uvnitř jedné CG). Pro zvýšení výkonu se UFS typicky vytváří s většími bloky (4 nebo 8 KB). Aby se pak neplýtvalo místem na malých souborech, umožňuje UFS uložit malý soubor do ne zcela využitého bloku na konci většího souboru (v terminologii UFS se tomuto říká *fragmenty*).

UFS původně používal synchronní zápis metadat, což v případě havárie počítače umožnilo programu `fsck` pro kontrolu disků hledat jen některé typy nekonzistencí a tím být rychlejší. Na druhé straně se v případě havárie v okamžiku zvětšování souboru mohlo stát, že informace o zvětšení je zanesena v metadatech, ale ještě ne v datech souboru, a že tedy soubor obsahuje nepřemazané bloky původně přidělené v jiných souborech (bezpečnostní problém). Novější implementace UFS umožňují asynchronní zápis metadat nebo asynchronní zápis metadat se zachováním pořadí operací (*soft updates*).

## 2.4 Ext2FS

Ext2FS (second extended filesystem) je souborový systém v Linuxu. Strukturou je podobný UFS – jen jednotlivé části se zde jmenují *block groups* a nejsou zarovnány podle fyzické geometrie disku jako u UFS (ostatně u dnešních disků se

zónovým zápisem ani informaci o fyzické geometrii nemáme), ale tak, aby pokud možno jednotlivé datové struktury přesně zaplnily celočíselný počet diskových bloků. Oproti UFS je zde navíc bitmapa volných i-uzlů, která slouží k rychlejšímu nalezení volného i-uzlu při vytváření nového souboru.




Obrázek 6 Struktura souborového systému Ext2FS

Ext2FS neimplementuje fragmenty a tento problém řeší opačným směrem: vytváří se s menší velikostí bloku (dříve 1 KB, dnes obvykle 4 KB) s tím, že při zápisu do souboru se bloky alokují po osmi (jsou-li k dispozici) a při uzavření souboru se nevyužité bloky uvolní. Tím se dosahuje podobné fragmentace jako u UFS s osminásobně většími bloky.

Mezi zajímavé vlastnosti Ext2FS patří tzv. rychlé symbolické linky – text symbolického linku, je-li kratší než 64 bajtů, je uložen přímo v i-uzlu na místě odkazů na datové bloky a dalších položek, které se u symbolických linků nevyužívají. Dále pak nové atributy souborů: nezměnitelný soubor, soubor pouze pro připojování dat na konec a bezpečné smazání souboru. Svazek může být připojen s asynchronním nebo synchronním zápisem metadat. Je možno zapnout nebo vypnout aktualizaci časů přístupu k souborům (časy modifikace se aktualizují i nadále).

# 3 Souborové systémy v Linuxu v současnosti

Většina současných souborových systémů přidává další vlastnosti: žurnálování (viz dále), access control lists (ACL), rozšířené atributy (například pro uložení bezpečnostního kontextu pro SELinux), často i on-line zvětšení svazku, například existuje-li nad LVM[3].

## 3.1 Tolerance k výpadku

Jedním z hlavních problémů starších souborových systému byla špatná tolerance k výpadku. I když program `fsck` je schopen nekonzistence opravit, jeho spuštění stojí čas a prodlužuje dobu výpadku. Souborové systémy se snaží podobným

[3]Logical Volume Manager

problémům předcházet podobným způsobem jako databáze – transakčním zpracováním. Uvažované změny jsou nejprve zapsány do logu (žurnálu) a teprve až je celá operace zapsána na disku v žurnálu, dochází ke změnám vlastních dat. V případě výpadku stačí jen přehrát ty transakce v žurnálu, které jsou zde zapsány jako kompletní.

Ani žurnálování ovšem nečiní program `fsck` zbytečným. V případech jako jsou hardwarové problémy a chyby operačního systému mohou i nadále vznikat nekonzistence souborového systému. Dostatečně robustní program pro kontrolu souborového systému je tedy i nadále podstatným kritériem výběru souborového systému.

Žurnálované souborové systémy obvykle transakčně zpracovávají jen operace nad metadaty. Důsledkem toho může být bezpečnostní problém podobného stylu, jako byl popsán u souborového systému UFS. Některé souborové systémy umožňují žurnálovat i operace nad daty, což ovšem s sebou nese větší režii. Jen velmi málo souborových systémů podporuje zlatou střední cestu – *uspořádané* (ordered) operace. V tomto režimu jsou zápisové operace nad daty vyřizovány asynchronně, ale teprve po jejich dokončení je do žurnálu ukončena transakce nad metadaty. Data samotná se tedy zapisují jen jednou (takže nedochází k významnému zpomalení proti pouhému žurnálování metadat), ale nemáme zmíněný bezpečnostní problém.

## 3.2 Ext3FS

Souborový systém Ext3FS je zpětně kompatibilním rozšířením Ext2FS. Nejvýznamnější změnou je žurnálování. Vývoj Ext3FS trval poměrně dlouho, nicméně jeho výsledkem je i obecná vrstva JBD (journalled block device), kterou používají i některé další souborové systémy a komponenty jádra. Žurnál může být umístěn i mimo vlastní blokové zařízení (například v baterií zálohované paměti RAM). Při korektním odpojení svazku (bez nedokončených transakcí v žurnálu) může být Ext3 svazek připojen jako Ext2.

Mezi další vlastnosti patří tzv. *sparse superblocks* – superblok není v každé block group, ale jen v exponenciálně narůstajícím intervalu BG, což sníží počet kopií superbloku na velkých svazcích a tím sníží dobu potřebnou pro připojení svazku i pro kontrolu svazku.

Adresáře v Ext3 mohou být uloženy jako lineární seznam nebo jako B-strom (což je efektivnější pro extrémně velké adresáře).

Ext3 implementuje všechny tři výše popsané režimy žurnálování s tím, že implicitní je *ordered* režim.

### 3.3 XFS

Do Linuxu se dostala implementace souborového systému XFS od SGI. Jedná se o plně 64-bitový žurnálovaný souborový systém. Svazek je rozdělen na oblasti velké obvykle 0,5 GB až 4 GB, nazývané *allocation groups* (AG). Informace v rámci AG jsou ukládány ve formě B+ stromu.

Při vytváření svazku lze uvést (případně `mkfs` zjišťuje sám), z kolika fyzických disků se skládá blokové zařízení a tuto informaci uloží do superbloku. Pomocí ní se pak v jádře počítá možná paralelizace diskových operací.

Asi nejzajímavější vlastností XFS je odložená alokace: většina souborových systémů používá odložený zápis dat (write-back). XFS toto posunuje ještě dále: u zapisovaných dat se i jejich *umístění* určuje až v okamžiku zápisu na disk. To umožní souborovému systému spojit více zápisových operací do jedné a dále tak snižovat fragmentaci. Na druhé straně toto komplikuje případnou implementaci *ordered* režimu pro žurnálování, takže XFS podporuje jen ostatní dva režimy. I zde je možno žurnálovat na externí zařízení.

XFS nepoužívá strukturu i-uzlu z obrázku 1. Jedná se o tzv. *extent-based* souborový systém: i-uzel obsahuje informace o jednotlivých úsecích (extents) souboru ve formě B-stromu. Toto umožňuje nižší režii u extrémně velkých souborů (nejsou-li příliš fragmentovány).

### 3.4 ReiserFS

Jedná se o souborový systém, který je celý organizovaný ve formě B+ stromu. Byl to vůbec první žurnálovaný souborový systém v Linuxu. Jeho nejzajímavější vlastností je, že alokační jednotkou je jeden bajt, nikoliv jeden sektor. Čili malé soubory zabírají jen nejnutnější místo, bez zarovnávání na velikost sektoru nebo dokonce nějakého většího bloku.

V případě problému prochází `reiserfsck` celé blokové zařízení a hledá na něm signatury uzlů B+ stromu. Což způsobí problémy, pokud například na ReiserFS uložíme soubor, ve kterém je obraz nějakého ReiserFS.

ReiserFS se nicméně již dále nevyvíjí (autoři pracují na vývoji Reiser4) a mimo jiné nepodporuje rozšířené atributy, včetně bezpečnostních kontextů pro SELinux. Nicméně pro netypické zátěže (malé soubory, velké množství souborů v jednom adresáři) je i nadále použitelný.

### 3.5 JFS

Od firmy IBM pochází souborový systém JFS (Journalled File System), původně implementovaný pro operační systém AIX. Jedná se opět o *extent-based* souborový systém se všemi základními vlastnostmi, které jsou pro Linux potřeba. Některé uživatele možná zarazí, že se žurnál vytváří s velikostí danou pevným

podílem z celkové velikosti svazku a tedy na extrémně velkých svazcích může opětovné připojení svazku s JFS (a přehrání transakcí v žurnálu) trvat poměrně dlouho.

## 3.6 JFFS2

Samostatnou třídou souborových systémů jsou systémy pro solid-state paměti, jako je třeba NAND flash paměť. Základní vlastností flash paměti je, že není třeba (na rozdíl od disků) optimalizovat na sekvenční přístup. Dále pak přepisovatelnost je omezená (průměrná životnost paměťového bloku je $10^4$ až $10^6$ přepsání). Velikost bloku je větší než u disků (i 32 KB). Na rozdíl od disku, který poskytuje dvě základní operace – čtení bloku a zápis do něj – má flash paměť operace tři: čtení, zápis a vymazání bloku. Před dalším zápisem je třeba blok vymazat. Často mají paměťové bloky k sobě ještě několik bajtů tzv. *out-of-band* dat navíc. Sem lze ukládat další informace o stavu konkrétního bloku.

Těchto zařízení je několik typů: většina USB paměťových karet má v sobě integrovaný paměťový řadič, který interně řeší překlad adres bloků tak, aby bylo zajištěno rovnoměrné opotřebení i při častém zápisu na jedno místo (*wear levelling*). Tyto paměťové karty pak mohou být použity i se souborovým systémem určeným pro disky. Paměťová zařízení která nemají integrovaný řadič s překladem adres (FTL, Flash Translation Layer) vyžadují speciální souborový systém. Běžné souborové systémy by totiž takovouto paměť častým přepisováním téhož místa (tabulka FAT, tabulka i-uzlů, žurnál apod.) zničily.

Souborové systémy pro flash paměti obvykle nepřepisují datové bloky na místě, ale vytvoří „novou verzi“ datového bloku. Při opětovném připojení svazku je pak třeba zkoumat, který fyzický blok obsahuje nejnovější verzi příslušného datového bloku.

Jedním z aktuálně používaných souborových systémů pro flash paměti je JFFS2 (Journalled Flash File System, verze 2). Jde o žurnálovaný souborový systém se stromovou strukturou, používající mechanismus garbage collection pro opožděnou recyklaci bloků, k nimž existuje novější verze. Jeho nevýhodou je, že při připojení musí projít celou flash paměť, protože metadata souborového systému si drží v paměti RAM počítače. Jak roste kapacita dostupných flash pamětí, začíná tato vlastnost JFFS2 více vadit.

## 3.7 OCFS2

Souborový systém OCFS2 (Oracle Cluster File System) byl původně vyvinut firmou Oracle pro jejich distribuovaný databázový stroj (RAC, Real Application Cluster). OCFS2 je používán poněkud jinak než ostatní souborové systémy: nepředpokládá se zde výlučný přístup počítače k příslušnému blokovému zařízení. Lze tedy jeden diskový prostor (například diskové pole přes síť SAN – storage

area network) připojit jako lokální souborový systém na více počítačích spojených do clusteru. OCFS2 si pak sám řídí komunikaci přes síť i přes toto blokové zařízení, aby pohled více strojů na tentýž svazek byl konzistentní.

Poznamenejme, že pro vyzkoušení OCFS2 nepotřebujeme mít drahé diskové pole a SAN – existují například víceportové disky s rozhranim IEEE 1394, případně můžeme vytvořit distribuovaný systém tolerantní k výpadku pomocí „síťového RAIDu“ zvaného DRBD.[4]

### 3.8 GFS2

Global File System je projekt zaměřený podobně jako OCFS2, to jest jako souborový systém clusterů se sdíleným diskovým polem. Z projektu GFS2 mimo jiné pochází zamykací nástroj DLM (distributed lock manager), který interně používá nejen GFS2, ale i OCFS2 a další subsystémy.

Příbuzným projektem GFS2 je také CLVM – cluster LVM, tedy nástroj pro koherentní operace nad volume managerem, který pracuje nad blokovým zařízením sdíleným z více počítačů.

## 4 Kam směřuje vývoj?

Vývoj v oblasti souborových systémů ale nekončí. V současné době ve světě Linuxu probíhají práce na několika nových souborových systémech. Některé z nich jsou už v relativně použitelném stavu a je možné je testovat. U jiných zatím není jasné, kde až se vývoj zastaví.

### 4.1 Ext4FS

Souborový systém ext4[5] je následníkem osvědčeného a robustního ext3. Zachovává diskový formát, i když kompatibilní je jen dopředně, nikoli zpětně: svazek ext3 lze připojit jako ext4dev, ale v případě použití nových vlastností ext4dev už není jednoduchá cesta zpět. Výhodou diskového formátu těchto souborových systémů (i včetně UFS, který je na tom podobně) je, že metadata souborového systému jsou víceméně na pevně definovaných místech. Tedy i v případě většího poškození souborového systému je šance na dohledání aspoň nějakých dat. U souborových systémů organizovaných jako B-stromy tuto možnost nemáme – metadata můžou být v podstatě kdekoli a jejich pozice na disku se i v čase mění.

[4] Distributed Replicated Block Device, `www.drbd.org`. Pomocí DRBD lze dvě diskové oblasti na dvou různých počítačích zrcadlit a na obou zpřístupnit jako blokové zařízení. S použitím OCFS2 pak může toto zařízení být na obou počítačích i používáno zároveň ve formě souborového systému.

[5] V současné době již dostupný v oficiálním jádře Linuxu pod jménem ext4dev.

Hlavním rozšířením ext4dev je volitelné zavedení *extent-based* struktury v i-uzlu namísto struktury z obrázku 1. Toto umožní efektivní ukládání velkých málo fragmentovaných souborů a přinese celkové zrychlení pro tyto soubory. Dále implementuje opožděnou alokaci místa pro zápis až v okamžiku zápisu na disk (podobně jako má XFS), časová razítka s rozlišením nanosekund (ext3 a většina dalších souborových systémů má rozlišení sekundové). Odstraňuje limit 32000 podadresářů v jednom adresáři, zavádí kontrolní součet žurnálu pro případ datové chyby.

Poměrně zajímavou vlastností jsou neinicializované *block groups*: v průběhu kontroly svazku programem `e2fsck` se totiž za normálních okolností musí projít tabulky i-uzlů a další struktury v každé BG. Pokud ale nejsou všechny i-uzly využívány, je to zbytečné zdržení. Ext4 si tedy pamatuje, až pokud byla daná struktura uvnitř BG nejdále použita, a tím pádem tato struktura (tabulka i-uzlů, bitmapa volných datových bloků, atd.) nemusí být ani celá inicializována při vytváření svazku programem `mke2fs`, ani celá kontrolována uvnitř `e2fsck`.

Pracuje se i na dalších rozšířeních jako je on-line defragmentace nebo podpora souborů větších než 2 TB.

Souborový systém Ext4 bude dostupný například v distribuci Fedora 9, která bude zveřejněna v květnu 2008.

## 4.2 Reiser4

Následníkem ReiserFS je souborový systém Reiser4. Jeho architektura je velmi podobná jako ReiserFS (včetně alokace místa až na úroveň bajtů), ale přináší některé revoluční vlastnosti. Některé z nich ale způsobují nekompatibilitu s normou POSIX, což vzbuzuje pochybnosti o zařazení Reiser4 do oficiálního jádra.

Základní vlastností Reiser4 je modularita. Souborový systém sám je v podstatě jen varianta B+ stromu na disku s tím, že různé činnosti jsou realizovány pomocí pluginů. Můžeme tak mít například plugin, který pro nově uložené soubory automaticky zajistí jejich indexování vyhledávacím softwarem, plugin pro kompresi a podobně. Soubor může mít více *proudů dat* (stream). Kromě hlavního proudu třeba proud s rozšířenými atributy. Každý soubor je pak vlastně i adresář (svých vlastních proudů), což je právě nepříliš konzistentní s POSIXem.

Reiser4 zpřístupňuje své transakční vlastnosti i pluginům a plánuje se i zpřístupnění aplikacím, takže uživatelský proces bude moci například udělat několik nezávislých operací nad soubory a jejich daty a pak buďto celou tuto sadu změn najednou provést (*commit*) nebo vrátit zpět (*rollback*).

Bohužel vývoj tohoto souborového systému v poslední době příliš nepokračuje[6] a tak není jisté, jestli se vůbec tohoto souborového systému v oficiálním jádře dočkáme.

[6] Hans Reiser byl v dubnu 2008 odsouzen za vraždu.

## 4.3 BTRFS a CRFS

BTRFS je dalším projektem firmy Oracle v oblasti souborových systémů Linuxu. Jde o souborový systém s kontrolními součty (podobně jako třeba Sun ZFS) a copy-on-write sdílením dat. Používá *extent-based* strukturu i-uzlu, nemá pevně alokovanou tabulku i-uzlů (alokace je až při vytvoření souboru). Je integrován s device mapperem v jádře pro zajištění funkce nad více zařízeními. Umožňuje mít v rámci jednoho svazku více kořenových adresářů (například pro atomické snímky – *snapshots* – svazku v určitém čase). Zajímavou vlastností BTRFS a jeho snímků je, že díky copy-on-write může být kterýkoli snímek i zapisovatelný.

BTRFS je možno začít používat bez reinstalace nad ext3fs – podporuje režim, kdy se ext3 svazek připojí jako BTRFS a teprve postupně dojde k migraci formátu metadat.

Paralelně s BTRFS je vyvíjen projekt CRFS – cache-koherentní síťový souborový systém. Jeho snahou je dosáhnout plně POSIXové sémantiky (na rozdíl od široce používaného NFS) při intentizvním cachování dat na straně klienta. CRFS vyžívá vlastností BTRFS – předpokládá se, že CRFS bude použitý jako server a síťový protokol pro zpřístupňování BTRFS svazků po síti.

## 4.4 POHMELFS

Alternativním projektem k CRFS je POHMELFS[7] Jevgenije Pojlakova. Podobně jako CRFS je v začátcích svého vývoje, i když je podle všeho o něco dále než CRFS.

Snahou autora je mít síťový souborový systém s více servery a možností odpojeného používání, jen nad lokální cache.

## 4.5 UBIFS

Pokračováním vývoje v oblasti specializovaných souborových systémů pro solid-state paměti je UBIFS. Používá překladovou (FTL) vrstvu UBI, která již je v jádře v subsystému Memory Technology Devices zahrnuta. UBIFS nad touto vrstvou staví běžný UNIXový souborový systém. Na rozdíl od JFFS2 má strukturu plně uloženou na disku, takže jeho připojení nevyžaduje procházení celé paměti.

Mezi další vlastnosti patří opožděný zápis (write-back), komprese při ukládání dat, žurnálování, možnost synchronních operací. Má dva režimy odpojení svazku: rychlé odpojení, kdy je po připojení potřeba přehrát transakce v žurnálu a normální odpojení, které je pomalejší, ale následné připojení je pak rychlejší, protože seznam neprovedených transakcí je již prázdný.

[7] Oficiální výklad této zkratky je Parallel Optimized Host Message Exchange Layered File System.

UBIFS je již téměř ve stavu, kdy je nasaditelný v produčním prostředí.

# 5 Závěr

Je tedy vidět, že vývojáři souborových systémů v Linuxu jsou stále aktivní a v brzké době můžeme očekávat zajímavé výstupy z několika probíhajících projektů v této oblasti.