07-08/2001

Linuxové noviny

Úvodem

David Häring

Doba prázdnin a letních dovolených je pomalu za námi a všichni zase zvolna přivykáme pracovnímu procesu. Ani v redakci LN nezaháláme, a tak tady máme letos již čvrté vydání Linuxových novin. Kromě vyhlášení vítězů prázdninové soutěže, kterou jsme vypsali v minulém čísle, v tomto čísle najdete recenzi knížky o MySQL, vydanou nakladatelstvím Computer Press, rozsáhlý článek o softwarové implementaci diskových polí RAID v Linuxu, návod jak instalovat Red Hat Linux a Windows95 na jeden disk, recenzi nedávno vydané linuxové verze slovníku Lingea Lexicon a popis obslužného software záložních zdrojů od APC — balíčku **Apcupsd**. Připomínky, náměty a zejména příspěvky :-) můžete jako obvykle posílat na adresu redakce (1). ■

1 adresa redakce
mailto:noviny@linux.cz

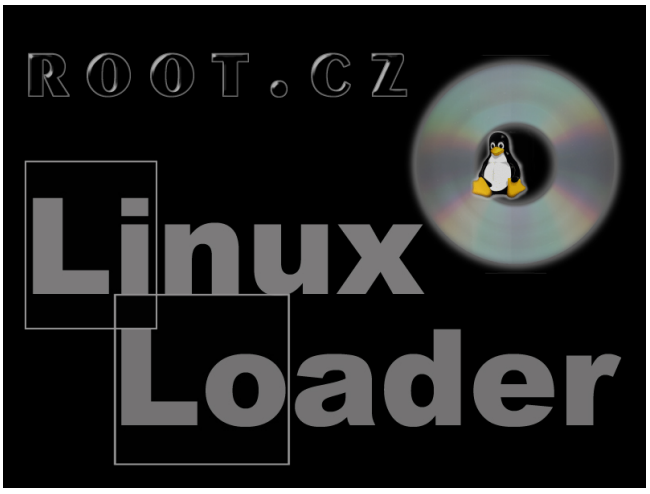
Soutěž Linuxových novin

Pavel Janík, 4. září 2001

V minulém čísle Linuxových novin vyhlásila redakce soutěž o nejhezčí grafickou obrazovku pro program LILO. Soutěže se zúčastnilo a pravidlům soutěže vyhovělo celkem devět grafických návrhů.

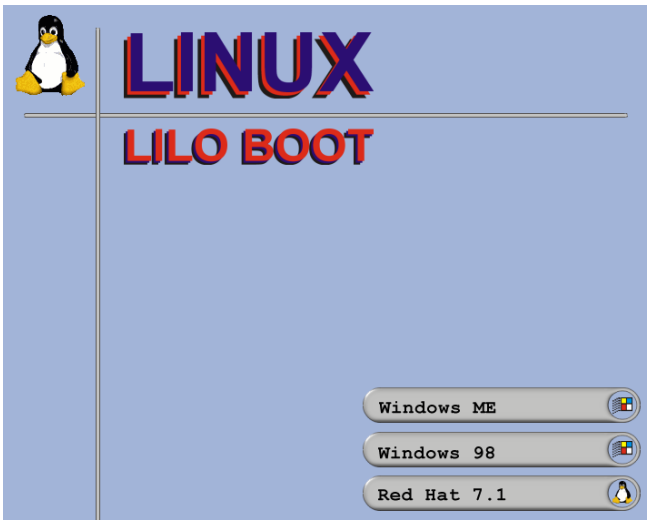
Odborná porota soutěže byla složena z redakce Linuxových novin (David Häring, Pavel Janík, Ondřej Vácha), Michala Krauseho (zástupce sponzora) a Dana Ohnesorga. Laická porota zasedala ve dvojici Miroslava Krátká a Marie Ohnesorgová.

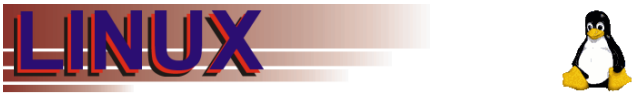
Velkým překvapením soutěže se stal grafický návrh Luboše Horni (luboss@bk.cz), který sice dopadl v hodnocení odborné i laické poroty nejlépe (získal celkem 13 bodů za



umístění od jednotlivých porotců, přičemž za první místo dostal návrh vždy tři body, za druhé dva a za třetí jeden), ale ve své podstatě není dokončen.

Na druhém a třetím místě se umístily návrhy od bruna@p14.mepnet.cz:





s dvanácti a pěti body. Vítěz soutěže obdrží od sponzora věcné dary a třeba se s podobným logem setkáme i v některé verzi linuxové distribuce…Všechny, nejen ty vítězné, návrhy grafických obrazovek pro LILO si můžete prohlédnout na adrese (1). ■

1 Návrhy grafických obrazovek pro LILO
ftp://ftp.linux.cz/pub/linux/local/noviny/2001-07/soutez

Linuxové noviny připravuje a vydává České sdružení uživatelů operačního systému Linux ISSN 1213-3655

Naučte se MySQL za 21 dní

Dan Ohnesorg, 21. července 2001

Pokud jste si teď řekli, co ten nám má co poroučet, vězte, že se nejedná o rozkaz, ale o název knihy. Vydalo ji nakladatelství Computer Press v edici Databáze.

Jedná se o překlad knihy nakladatelství Sams, které vydává celou edici knih, snažících se čtenářům podat přívětivou a srozumitelnou formou základy různých informačních technologií.

Konkrétně tato kniha se podle mého názoru povedla. Ačkoliv je úzce orientována na MySQL, vysvětluje základy SQL, smysl používání databází či normalizaci dat, a tak její první polovinu můžete použít bez úprav s jakoukoliv jinou databází. Druhá část již řeší konkrétní využití MySQL (mimo jiné i spojení s jazykem **Perl** či **PHP**, tvorbu **CGI** scriptů) a je méně přenositelná.

Jestli látku skutečně zvládnete za 21 dní nevím, ale je to celkem pravděpodobné. Předpokládají se alespoň základní vědomosti o počítačích, které jistě čtenář této recenze má, když už ji dokázal najít na Internetu. Výklad je rozhodně polopatický.

Zvláštní zmínku si zaslouží překlad a sazba. Sazba dostává pět mínus, protože příklady jsou sázeny fontem, který nemá jednoduché uvozovky a na jejich místě je vytištěn znak ‡, a tak začátečník, kterému je kniha určena, bude asi dost zmaten. Taktéž dlouhé názvy parametrů jsou zmršené, asi těžko se vám povede import provedený příkazem s parametrem `-fields-enclosed-by=‡ -fields terminated-by=/` (pokud nejste zběhlí v syntaxi, tak vězte, že oba parametry musí začínat dvěma mínusy a v druhém samozřejmě není mezera mezi slovy „fields“ a „terminated“, nýbrž je tam opět mínusek). Chyba je zákeřná v tom, že v knize se místo dvou mínusků vyskytují hned tři varinaty, skutečně správně vysázené dva mínusy, jeden krátký minus a dva mínusky slité do dlouhé pomlčky. (Pro rýpaly, ten příkaz má samozřejmě další problém, i kdyby v něm byl správně vytištěn jednoduchý apostrof, musel by se escapovat aby se k mysql dostal, ale protože nevím, jak je to v originální knize, tak nemohu obvinit z vady sazbu.)

Překlad je pěkný, dobře se čte, neobsahuje šroubované věty, se kterými se občas setkáváme. Ale je poněkud nepřesný, jeden příklad za všechny, parametr -h u `mysqldump` dělá prý tohle: „-h=názevhostitele. Vytvoří soubor výpisu na určeném hostiteli, místo na implicitně nastaveném místním hostiteli (localhost).“ My, kteří víme, že parametr udává název serveru, na kterém je uložena databáze se kterou pracujeme, jsme celkem v pohodě. Dokážeme si to už nějak schroustat. Začátečník ale bude marně hledat vydumpovaná data na serveru, kde nejsou a naopak nebude vědět, kam má napsat jméno serveru, se kterým pracuje. (Opět jen tak na okraj, v anglické dokumentaci je to popsáno prostě a jasně: -h|--host=hostname... Connect to host.)

Nicméně celkově se mi kniha líbila. Myslím si, že by Computer Press mohl vydat i další knihy z této edice a nezbývá než mu popřát více štěstí při finálních úpravách knihy.

Naučte se MySQL za 21 dní, Mark Maslakowski, vydalo nakladatelství a vydavatelství Computer Press v roce 2001. 480 stran, doporučená prodejní cena 490 Kč. Členové CZLUGu mohou knihu získat se slevou. ■

Softwarový RAID pod Linuxem

David Häring, 28. července 2001

Definice RAIDu

Na rozdíl od jiných hardwarových problémů jsou výpadky disků obvykle spojeny s časově náročnou obnovou dat ze záloh. Těmto problémům se ale můžeme vyhnout použitím redundantních diskových polí **RAID**, která jsou vůči výpadkům jednotlivých disků odolnější. Označení RAID pochází z anglického „Redundant Array of Inexpensive Disks“ nebo také „Redundant Array of Independent Disks“. Jedná se tedy o několik disků sloučených do jednoho logického svazku, který zpravidla zajišťuje určitou redundanci dat, díky které je pak odolný vůči hardwarovým výpadkům některého z disků. I když je v názvu obsaženo slovíčko „redundant“, ne všechny typy RAIDu jsou skutečně redundantní. Zatímco některé typy RAIDu jsou navrženy s ohledem na maximální bezpečnost dat, jiné jsou naopak optimalizovány na rychlost.

Hardwarové a softwarové implementace RAIDu

RAID lze provozovat v podstatě dvojím způsobem. Buď je realizován v hardwaru, což obnáší speciální řadič osazený procesorem a zpravidla vybavený vlastní pamětí, která slouží jako cache. Veškeré funkce RAIDu plní řadič a z pohledu operačnímu systému se chová jako jediný disk. Tato řešení bývají poměrně drahá (Mylex, DPT — nyní Adaptec, ICP Vortex, velcí výrobci PC jako HP, IBM, Compaq apod. mají své vlastní implementace). Předností hardwarových řešení bývá maximální spolehlivost a ve srovnání se softwarovou variantou RAIDu dovedou odlehčit zátěži systému. RAID ovšem také může být realizován patřičným ovladačem na úrovni operačního systému a spousta operačních systémů to také dnes umožňuje. Toto řešení může být za jistých okolností flexibilnější a rychlejší, ale také náročnější na systémové prostředky — zejména na čas procesoru. V posledních letech se setkáváme i s napůl hardwarovými/softwarovými implementacemi RAIDu, kdy hardware obsahuje jen minimální podporu a většinu práce dělá ovladač — tato varianta je levná, ale řada produktů této kategorie je nevalné kvality a výkonu. Tento článek je zaměřený na softwarový RAID pod Linuxem.

Teorie fungování RAIDu

Než se zaměříme na detaily softwarového RAIDu pod Linuxem, podíváme se na princip fungování jednotlivých typů RAIDu a jejich vlastnosti.

RAID 0 (Nonredundant striped array)

Tento typ je určen pro aplikace, které vyžadují maximální rychlost a není redundantní. Naopak je potřeba vzít v úvahu, že pravděpodobnost výpadku takového pole roste s počtem disků. Ideální použití představují audio/video streamingové aplikace, eventuálně databáze a obecně aplikace, při kterých čteme sekvenčně velká množství dat. Základní jednotkou pole je tzv. **stripe** (z angl. „stripe“, česky pruh), což je blok dat určité velikosti (běžně 4-64kB v závislosti na aplikaci). Po sobě jdoucí data jsou pak v poli rozložena střída-

vě mezi disky do „stripů“ takovým způsobem, aby se při sekvenčním čtení/zápisu přistupovalo ke všem diskům současně. Tím je zajištěna maximální rychlost jak při čtení tak i zápisu, ale současně je tím dána také zranitelnost pole. Při výpadku kteréhokoliv disku se stávají data v podstatě nečitelná (respektive nekompletní). RAID 0 bývá označován rovněž jako **striping**. Protože není redundantní, má nejvýhodnější poměr cena/kapacita. Počet disků je libovolný. Je ovšem třeba pamatovat na to, že s rostoucím počtem disků v poli roste i pravděpodobnost výpadku pole (protože výpadek libovolného disku znamená havárii celého pole); RAID 0 je tedy velmi rychlý, ale méně bezpečný než samostatný disk.

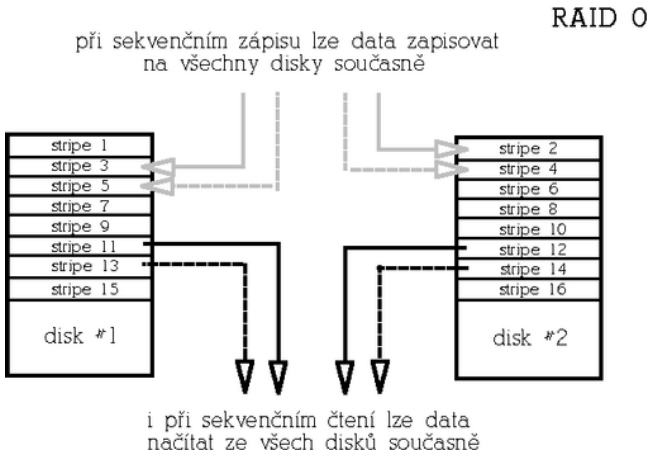


schéma pole RAID 0

RAID 1 (Mirrored array)

RAID 1 je naopak maximálně redundantní. Rychlost čtení může být oproti samostatnému disku výrazně vyšší, rychlost zápisu je stejná jako u samostatného disku. Funguje tak, že data jsou při zápisu „zrcadlena“ na všechny disky v poli (tedy v případě RAIDu 1 tvořeného dvěma disku jsou data duplikována apod.). Při čtení lze využít vícero kopií dat a podobně jako u RAIDu 0 číst za všech disků současně. Tento typ pole je určen pro aplikace s důrazem na maximální redundanci. Výhodou tohoto redundantního řešení je stabilní výkon i v případě výpadku disku, nevýhodou je poměr cena/kapacita. Počet disků bývá buď 2 anebo libovolný, čím větší počet disků, tím větší redundance a odolnost proti výpadku.

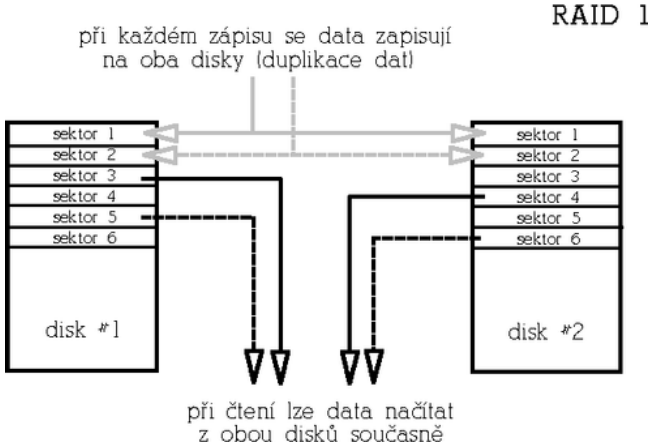


schéma pole RAID 1

RAID 2 (Parallel array with ECC)

Pole tohoto typu jsou dnes již historií, protože dnešní disky mají vlastní opravné mechanismy a uchovávají ECC informace pro každý sektor samy. V polích tohoto typu se stripovalo po jednotlivých sektorech a část disků pole byla vyhrazena pro ukládání ECC informací. Jakékoliv čtení i zápis proto zpravidla zahrnovalo přístup ke všem diskům pole, což bylo překážkou vyššího výkonu zejména u aplikacích pracujících se většími kusy dat. Tento typ není redundantní.

RAID 3 (Parallel array with parity)

Také tento typ polí se již nepoužívá, jedná se o předchůdce RAIDu 4. Stripovalo se po sektorech, ale jeden disk byl vyhrazen jako paritní, což zajišťovalo redundanci (princip zajištění redundance je stejný jako u RAIDu 4 a 5, který je popsán níže). Protože i v tomto případě se stripovalo po sektorech, jakékoliv čtení i zápis zpravidla zahrnovalo přístup ke všem diskům pole.

RAID 4 (Striped array with parity)

Raid 4 je redundantní pole, které se dnes již používá málo. Funkčně je podobné RAIDu 5, který je ale výkonnější. Funguje tak, že jeden disk je vyhrazen jako tzv. **paritní disk**. Na paritním disku je zaznamenán kontrolní součet (operace XOR přes data stejné pozice jednotlivých disků). Pokud tedy dojde k výpadku některého z datových disků, lze data rekonstruovat z dat zbylých disků a parity uložené na paritním disku. RAID 4 je odolný vůči výpadku libovolného jednoho disku a má tedy příznivý poměr cena/kapacita. Paritní disk ale představuje úzké hrdlo této architektury při zápisech, protože každý zápis znamená také zápis na paritní disk. Mimimální počet disků je 3.

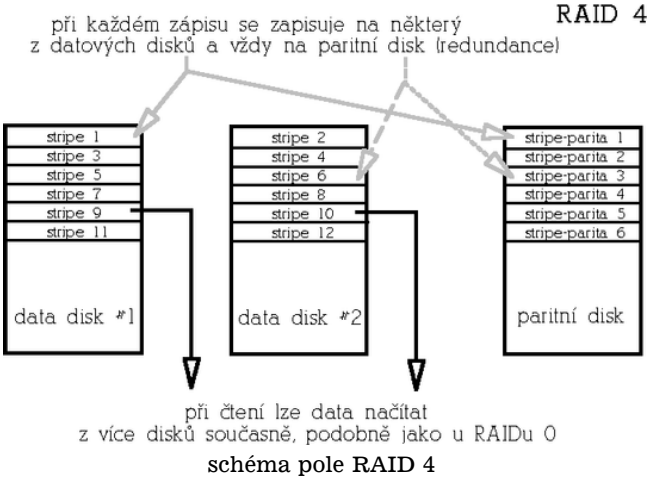


schéma pole RAID 4

RAID 5 (Striped array with rotating parity)

Tento typ poskytuje redundanci vůči výpadku libovolného jednoho disku s dobrým poměrem cena/kapacita a výkonem. RAID 5 je vylepšená varianta RAIDu 4 v tom, že **parita** není uložena na jednom vyhrazeném disku, ale **je rozmístěna rovnoměrně mezi všemi disky pole**, čímž se odstraní úzké hrdlo architektury RAIDu 4.

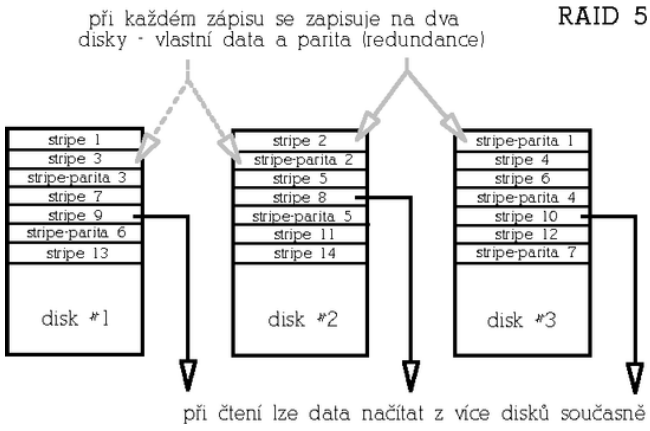


schéma pole RAID 5

Paritu lze spočítat buď tak, že skutečně načteme a XORujeme data z odpovídajících datových stripů všech disků (takto se parita počítá při inicializaci pole, vyžaduje to tedy přístup ke všem diskům). Ve druhém případě načteme původní data z datového stripu, která se mají změnit, provedeme XOR s novými daty a výsledek ještě XORujeme s původní hodnotou parity (takto se parita počítá na již inicializovaném běžícím poli). Zápis dat tedy představuje dvoje čtení (dat a parity), výpočet parity a dvojí zápis (opět dat a parity). Počet přístupů na disk při zápisu je v tomto případě konstantní bez ohledu na počet disků v poli — přistupuje se vždy ke dvěma diskům — a to také má za následek nižší výkon tohoto typu pole ve srovnání s redundantním RAIDem 1.

V **degradovaném režimu** (degradovaný režim znamená stav, kdy je některý z disků z pole vyřazen kvůli hardwarové chybě) se pak musejí data uložená na vadném disku odvodit z dat zbývajících disků a parity. Na rozdíl od redundantního RAIDu 1, kde výpadek disku obvykle neznamená výrazný pokles výkonu, vykazuje RAID 5 v degradovaném režimu výrazně snížený výkon zejména při čtení. Minimální počet disků pro tento typ diskového pole jsou 3.

Kombinace více typů polí

Z definic výše popsaných typů polí vyplývá, že redundantní pole nejsou tak rychlá, jak bychom si mohli přát a naopak u RAIDu 0 nám chybí redundance. Existuje ovšem možnost, jak výhody jednotlivých typů diskových polí spojit. Tato metoda spočívá ve vytvoření kombinovaných polí, kdy disky v poli určitého typu jsou samy tvořeny poli jiného typu. Příkladem může být např. RAID 1+0, kdy jsou pole typu RAID 1 dále sloučeny do RAIDu 0. Takové pole je pak redundantní (toleruje výpadek až dvou disků), rychlejší zejména v zápisech než samotný RAID 1, a má lepší poměr cena/kapacita než RAID 1 (velikost je zde n/2 * disk a minimální počet disků je pak 4. Další možností je třeba RAID 0+1, RAID 5+0, RAID 5+1 apod.

Jak RAID ošetří výpadek disku

Všechny typy redundantních polí obvykle umožňují nakonfigurovat kromě **aktivních disků** ještě 1 či více **rezervních disků**. Aktivními disky zde rozumíme disky, které jsou součástí funkčního pole. V případě výpadku některého z aktivních disků pak může systém místo vadného disku okamžitě začít používat disk rezervní. Po aktivaci rezervního disku do pole systém provede na pozadí (tedy bez narušení dostupnosti pole) **rekonstrukci** pole a jakmile je rekonstrukce hotova, je pole opět plně redundantní. Rekonstrukcí je míněna buď synchronizace obsahu nového disku s ostatními aktivními disky (v prípadě RAIDu 1), anebo rekonstrukce obsahu původního vadného disku na základě redundantní informace (jedná-li se o RAID 4 nebo 5). Po dobu, než rekonstrukce proběhne, se pole nachází v tzv. degradovaném režimu, kdy v závislosti na konfiguraci nemusí být redundantní a v případě RAIDu 5 se to projeví sníženým výkonem (odtud název „degradovaný režim“). Je samozřejmě možné pole provozovat bez rezervních disků a disk vyměnit později manuálně. Detailně se na výměnu disků a rekonstrukci polí ještě podíváme později.

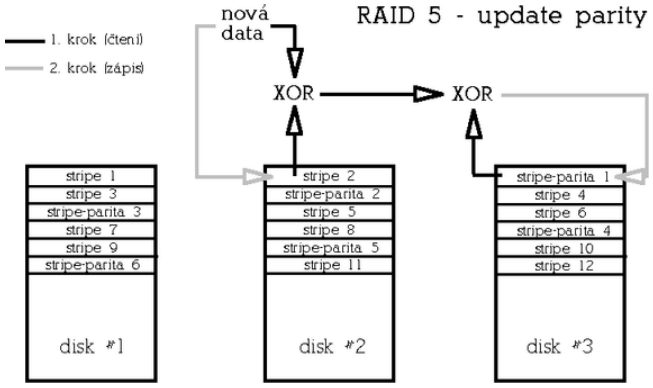
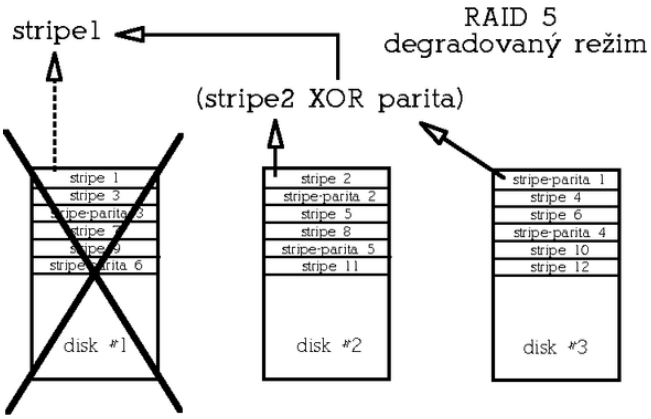


schéma výpočtu parity pro RAID 5

Redundantní pole neznamenají konec záloh

Redundantní disková pole jsou odolná pouze vůči výpadkům určitého počtu disků. Neochrání před výpadkem napájení, poškozením souborového systému při pádu celého systému nebo chybou administrátora. Proto je potřeba myslet i na další metody ochrany dat — např. na záložní zdroje napájení (UPS), žurnálovací souborové systémy apod. a v každém případě pravidelně zálohovat.



pole RAID 5 v degradovaném módu

Typy polí podporovaných Linuxovým ovladačem RAIDu

Až doposud jsme se zabývali pouze teorií fungování diskových polí, podívejme se tedy jak to vypadá se softwarovým RAIDem v Linuxu. Softwarová implementace RAIDu pod OS Linux podporuje 5 typy diskových polí:

Typ	počet disků	kapacita	redundance	shrnutí
Linear	2 a více	n x disk	ne	není raid, pouze sloučení disků
RAID 0	2 a více	n x disk	ne	maximální výkon při čtení i zápisu
RAID 1	2 a více	disk	ano	maximální redundance, bez výrazné degradace výkonu při výpadku, rychlejší čtení, vyšší cena
RAID 4	3 a více	(n-1) x disk	ano	nepoužívá se, nahrazen RAID 5
RAID 5	3 a více	(n-1) x disk	ano	výhodný poměr cena/kapacita

Výpis č. 1: shrnutí používaných typů SW RAIDu

- **Linear** — toto v podstatě není RAID, jedná se o možnost „pospojování“ více disků do jednoho. Co se týče rychlosti, neliší se příliš od výkonu samostatných disků. Není redundantní, výhodou však je např. oproti RAIDu 0 to, že se v případě havárie jednoho z disků dají data ze zbývajících disků snáze obnovit, protože se nestripuje.
- **RAID 0** — zde je volitelná velikost stripu; tato výrazně (v závislosti na aplikaci) ovlivňuje výkon RAIDu.
- **RAID 1** — oproti klasické definici RAIDu 1 kdy se disky spojují pouze do párů, pod Linuxem můžeme vytvořit RAID 1 i z více než dvou aktivních disků.
- **RAID 4** — je sice implementován, ale funkčně nahrazen RAIDem 5
- **RAID 5** — opět je zde volitelná velikost stripu, která ovlivňuje výkon RAIDu.

Historie, dostupnost, omezení

Původní ovladač „MetaDisku“ (odtud název ovladače — **md**) napsal Marc Zyngier. Implementace zpočátku delší dobu podporovala pouze typy linear, RAID 0 a RAID 1. Později ovladač vyvíjeli zejména Ingo Molnár, Gadi Oxman a nyní Neil Brown. Přibyla podpora dalších typů polí a podpora bootování z RAIDu 0 a 1. V současné době se můžeme setkat se dvěma verzemi ovladače: verzí 0.42, kterou obsahují čistá jádra řady 2.2.x a verzí 0.90, která se nachází v jádrech řady 2.4.x. Jádra 2.2.x většiny distribucí ale obsahují podporu RAIDu verze 0.90, ovladač této verze pro jádra 2.2.x je k dispozici ve formě záplaty jádra (1). K provozování softwarového RAIDu je zapotřebí kromě jaderného ovladače také sada utilit **raidtools** (dříve mdtools), které musejí odpovídat verzi jaderného ovladače (raidtools 0.42 (2) a raidtools 0.90 (3).

Ovladače RAIDu jsou testovány především na platformě x86, ale lze je provozovat i na platformách Sun či Alpha (a možná i dalších, ovšem na těchto exotičtějších platformách jsou méně odladěny). Pro jádra řady 2.2.x platí určitá omezení, která vyplývají z architektury této řady jader:

- na redundantním poli (tedy RAID 1 a 5) není možno bezpečně swapovat (a to ani do souboru) v průběhu rekonstrukce pole;
- na redundantní pole (tedy RAID 1 a 5) není možno provozovat žurnálovací souborový systém (lépe řečeno, pokud probíhá na RAIDu 1 rekonstrukce pole, nesmí se používat žurnálování, na RAIDu 5 žurnálování použít nelze vůbec);
- pole nelze libovolně kombinovat (přesněji lze vytvořit na první pohled funkční kombinované pole, ale výpadek disku může způsobit pád celého pole). Konkrétně je možné vytvářit pole 0+1 a kombinace polí 1,4 a 5. Nelze ale použít pole [145]+0.

Pro jádra řady 2.4 tato omezení neplatí.

Konfigurace

Konfigurace RAIDu verze 0.90 používá konfigurační soubor `/etc/raidtab` (na rozdíl od verze 0.42, kde byla konfigurace uložena v `/etc/mdtab`), ve kterém se používají následující direktivy:

- `raiddev`: touto direktivou definice pole začíná, následuje označení pole. Svazky softwarového RAIDu se označují `md1` až `mdX`.
- `raid-level`: následuje direktivu `raiddev`, uvádíme zde typ pole (-1 = linear, 0 = RAID 0, 1 = RAID 1, 5 = RAID 5).
- `persistent-superblock`: tato direktiva bude popsána níže, týká se kompatibility se starší verzí RAIDu.
- `chunk-size`: velikost stripu, maximální velikost je 4MB (což je dáno konstantou `MAX_CHUNK_SIZE` ovladače), udává se v kB.
- `nr-raid-disks`: zde uvádíme kolik diskových oddílů bude součástí pole (maximální počet je v jádrech 2.2.x 12, ale jsou k dispozici záplaty, které tento limit podstatně zvyšují)
- `nr-spare-disks`: počet rezervních disků v poli
- direktiva `device jméno_oddílu` následovaná jednou z direktiv `raid-disk`, `spare-disk`, `parity-disk` nebo `failed-disk`: tyto direktivy deklarují příslušné oddíly, které budou součástí pole.
- `raid-disk`: tento oddíl bude aktivním oddílem.
- `spare-disk`: tento oddíl bude sloužit jako rezervní.
- `parity-disk`: tento oddíl bude sloužit jako paritní disk (RAID 4).
- `failed-disk`: tento oddíl bude při inicializaci pole pře-

skočen (má význam pouze při sestavování pole v degradovaném stavu, viz diskuse níže).

- `parity-algorithm`: specifikuje schéma rozložení parity u RAIDu 5 (možnosti jsou: `left-symmetric`, `right-symmetric`, `right-asymmetric`; z podporovaných je obecně nejrychlejší `left-symmetric`).

Konfigurace softwarového RAIDu je popsána v příslušném dokumentu HOWTO (4).

Starší verze softwarového RAIDu (0.40 až 0.51) používaly konfigurační soubor `/etc/mdtab`. Přestože tato starší verze ovladače je součástí „čistých jader“ 2.2.x, je zastaralá a nebudeme se jí zde vůbec zabývat. Pro úplnost je starší verze RAIDu dokumentovaná v příslušném HOWTO (5).

Příklady konfigurací

Příklad konfigurace pole **linear**. Mějme pole typu linear složené ze dvou oddílů:

```
raiddev                 /dev/md0
raid-level              -1
persistent-superblock   1
nr-raid-disks           2
nr-spare-disks          0
device                  /dev/sda1
raid-disk               0
device                  /dev/sdb1
raid-disk               1
```

Příklad konfigurace **RAID 0**. Mějme diskové pole RAID 0 složené ze dvou oddílů, sda1 a sdb1, velikost stripu je 16 kB:

```
raiddev                 /dev/md0
raid-level              0
persistent-superblock   1
chunk-size              16
nr-raid-disks           2
nr-spare-disks          0
device                  /dev/sda1
raid-disk               0
device                  /dev/sdb1
raid-disk               1
```

Příklad konfigurace **RAID 1**. Mějme diskové pole RAID 1 složené ze dvou oddílů a s jedním rezervním oddílem:

```
raiddev                 /dev/md1
raid-level              1
nr-raid-disks           2
nr-spare-disks          1
device                  /dev/sda1
raid-disk               0
device                  /dev/sdb1
raid-disk               1
device                  /dev/sdc1
spare-disk              0
```

Příklad konfigurace pole **RAID 5** s rozložením parity „left-symmetric“, velikostí stripu 4 kB a jedním rezervním oddílem:

```
raiddev                 /dev/md2
raid-level              5
nr-raid-disks           3
chunk-size              4
parity-algorithm        left-symmetric
nr-spare-disks          1
device                  /dev/sda1
raid-disk               0
device                  /dev/sdb1
raid-disk               1
device                  /dev/sdc1
raid-disk               2
device                  /dev/sdc1
spare-disk              0
```

Obslužný software — Raidtools

Balíček **raidtools** (nahrazuje starší balíček **mdtools** určený pro starší verze ovladače RAIDu) obsahuje obslužné utility nezbytné k manipulaci s diskovými poli:

- `mkraid`: pro inicializaci polí;
- `raidstart`: pro spouštění diskových polí;
- `raidstop`: pro vypnutí diskových polí;
- `raidhotadd`: přidá nový diskový oddíl do aktivního diskové pole (náhradou za vadný oddíl, pokud jsou všechny oddíly pole funkční, přidá nový oddíl jako rezervní — „spare-disk“). Nelze tedy použít pro rozšíření kapacity pole;
- `raidhotremove`: odejme vadný diskový oddíl z aktivního diskového pole;
- `raidsetfaulty`: označí funkční diskový oddíl jako vadný, tím umožní jeho odejmutí z pole příkazem `raidhotremove` (možné využití např. při testování nebo výměnách funkčních disků); v posledních distribucích Red Hat se pro tento účel používá utilita `raidhotgenerateerror`;
- `raid0run`: utilita pro spouštění starších polí typu linear nebo RAID 0 bez perzistentních superbloků (viz níže „Perzistentní superbloky a RAID 0 / linear“).

V budoucnu zřejmě budou utility z balíčku **raidtools** nahrazeny jedinou utilitou **mdctl**, kterou vyvíjí Neil Brown. První verze této utility (6) jsou již k dispozici. Utilita `mdctl` nemusí používat žádný konfigurační soubor, vše potřebné lze zadat na příkazové řádce, anebo to `mdctl` zjistí analýzou RAID superbloků uložených na discích (podobně funguje automatické startování polí jádrem při bootu). Cílem autora je tedy přidat výhody a robustnost, kterou poskytuje vlastnost **raid autodetect** jádra a zároveň se vyhnout potenciálním konfliktům mezi neaktuální konfigurací v souboru `raidtab` a skutečnou konfigurací polí, ke kterým časem může dojít, pokud používáme **raidtools**. Zatím je utilita `mdctl` ve stavu vývoje, takže její ostré použití ještě nelze doporučit.

Poznámka na vysvětlenou: Soubor `raidtab` odráží konfiguraci polí v době jejich sestavení, ovšem pokud třeba později vyměníme nebo přesuneme některé disky, nemusí již odrážet skutečnou konfiguraci. Pokud tedy z nějakého důvodu potřebujeme pole znovu inicializovat anebo ho jen startujeme pomocí `raidstart`, nesmíme zapomenout soubor `raidtab` ručně upravit, abychom se ušetřili v budoucnu nepříjemností.

Inicializace polí

Jakmile máme odpovídajícím způsobem rozdělené disky

```
Disk /dev/sda: 255 heads, 62 sectors, 1124 cylinders
Units = cylinders of 15810 * 512 bytes

   Device Boot    Start       End    Blocks   Id  System
/dev/sda1             1        17    134354   fd  Linux raid autodetect
/dev/sda2            18        51    268770   fd  Linux raid autodetect
/dev/sda3            52       118    529635   fd  Linux raid autodetect
/dev/sda4           119      1124   7952430    5  Extended
/dev/sda5           119       152    268739   82  Linux swap
/dev/sda6           153       347   1541444   fd  Linux raid autodetect
/dev/sda7           348      1124   6142154   fd  Linux raid autodetect
```

Výpis č. 2: výpis fdisku — nastavení diskových oddílů pro autodetekci RAID polí

a připravený konfigurační soubor `/etc/raidtab`, můžeme pole inicializovat utilitou `mkraid`, která pole sestaví a aktivuje. Pokud zakládáme pole s perzistentními RAID superbloky (viz níže), pak `mkraid` vypíše i pozici RAID superbloků:

```
# mkraid  /dev/md5

handling MD device /dev/md5
analyzing super-block
disk 0: /dev/sda7, 20163568kB,\
 raid superblock at 20163456kB
disk 1: /dev/sdb7, 20163568kB,\
 raid superblock at 20163456kB
```

Po úspěšné inicializaci bychom měli v souboru `/proc/mdstat`, který obsahuje informace a aktivních polích vidět odpovídající záznam, např:

```
$ cat /proc/mdstat

Personalities : [raid0] [raid1] [raid5]
read_ahead 1024 sectors
md0 : active raid1 sdb1[1] sda1[0]\
 131968 blocks [2/2] [UU]
```

Poté nám nic nebrání pole zformátovat např. pomocí `mke2fs`, pokud chceme na poli provozovat souborový systém `ext2`. Utilita `mke2fs` akceptuje volbu `-R stride=X`, která udává kolik bloků souborového systému obsahuje 1 „stripe“ pole. Tím pádem je také vhodné zadat ručně velikost bloku (parametr `-b`). Např. mějme pole typu RAID 0 s velikostí stripu 16 kB. Pokud budeme chtít použít velikost bloku souborového systému 4 kB, zadáme:

```
mke2fs /dev/md0 -b 4096 -R stride=4
```

Raid autodetect anebo raidstart?

Nyní tedy máme funkční diskové pole. Zbývá vyřešit způsob, jakým se bude pole vypínat při vypnutí systému a zapínat při startu systému. Jednou možností je použití utilit `raidstart` a `raidstop`. Pomocí těchto utilit můžeme pole aktivovat či zastavit kdykoliv, stačí tedy upravit příslušné startovací skripty. (Pokud už distribuce toto neobsahuje; např. distribuce Red Hat není potřeba upravovat, ze skriptu `/etc/rc.d/rc.sysinit` je `raidstart` volán automaticky, existuje-li soubor `/etc/raidtab` a `raidstop` je volán ze skriptu `/etc/rc.d/init.d/halt`.).

Druhou, robustnější metodou je využití možnosti automatické aktivace polí jádrem při bootu. Aby mohla fungovat, je potřeba v prvé řadě používat perzistentní RAID superbloky a všechny diskové oddíly, které jsou součástí polí, musejí být v tabulce oddílů označeny jako typ `Linux raid autodetect` (tedy hodnota `0xfd` hexadecimálně). Viz výpis fdisku — nastavení diskových oddílů pro autodetekci RAID polí.

Výhodou tohoto řešení navíc je, že jakmile je diskové pole inicializováno, nepoužívá se již pro opětovný start / zastavení pole konfigurační soubor `/etc/raidtab`. O sestavení a spuštění pole se postará ovladač RAIDu, který na všech diskových oddílech typu `Linux raid autodetect` vyhledá RAID superbloky a na základě informací v RAID superblocích pole spustí. Stejně tak ovladač RAIDu všechny pole korektně vypne v závěrečné fázi ukončení běhu systému poté, co jsou odpojeny souborové systémy. I v případě změny jmen disků nebo po přenesení disků na úplně jiný systém tedy pole bude korektně sestaveno a nastartováno, viz následující úryvek systémového logu po zámeně disků `sdc` za `sdb` a `sdb` za `sda`:

```
autorun ...
considering sdb6 ...
  adding sdb6 ...
  adding sda6 ...
created md4
bind<sda6,1>
bind<sdb6,2>
running: <sdb6><sda6>
now!
sdb6's event counter: 00000001
sda6's event counter: 00000001
md: device name has changed from sdc6 to sdb6\
 since last import!
md: device name has changed from sdb6 to sda6\
 since last import!
md4: max total readahead window set to 128k
raid1: device sdb6 operational as mirror 1
raid1: device sda6 operational as mirror 0
md: updating md4 RAID superblock on device
sdb6 [events: 00000002](write) sdb6's\
 sb offset: 15332224
sda6 [events: 00000002](write) sda6's\
 sb offset: 15332224
```

RAID superblok

Každý diskový oddíl, který je součásti raid svazku (výjimku tvoří pouze svazky bez perzistentních superbloků, viz níže) obsahuje tzv. **RAID superblok**. Tento superblok je 4kB část RAID oddílu vyhrazená pro informace o příslušnosti daného oddílu k určitému poli a o stavu pole. Následuje podrobný

```
/* Device "operational" state bits */
#define MD_DISK_FAULTY          0 /* disk is faulty / operational */
#define MD_DISK_ACTIVE          1 /* disk is running or spare disk */
#define MD_DISK_SYNC            2 /* disk is in sync with the raid set */
#define MD_DISK_REMOVED         3 /* disk is in sync with the raid set */

typedef struct mdp_device_descriptor_s {
        __u32 number;           /* 0 Device number in the entire set          */
        __u32 major;            /* 1 Device major number                      */
        __u32 minor;            /* 2 Device minor number                      */
        __u32 raid_disk;        /* 3 The role of the device in the raid set   */
        __u32 state;            /* 4 Operational state                        */
        __u32 reserved[MD_SB_DESCRIPTOR_WORDS - 5];
} mdp_disk_t;

#define MD_SB_MAGIC             0xa92b4efc
/* Superblock state bits */
#define MD_SB_CLEAN             0
#define MD_SB_ERRORS            1

typedef struct mdp_superblock_s {
        /* Constant generic information */
        __u32 md_magic;         /*  0 MD identifier                           */
        __u32 major_version;    /*  1 major version to which the set conforms */
        __u32 minor_version;    /*  2 minor version ...                       */
        __u32 patch_version;    /*  3 patchlevel version ...                  */
        __u32 gvalid_words;     /*  4 Number of used words in this section    */
        __u32 set_uuid0;        /*  5 Raid set identifier                     */
        __u32 ctime;            /*  6 Creation time                           */
        __u32 level;            /*  7 Raid personality                        */
        __u32 size;             /*  8 Apparent size of each individual disk   */
        __u32 nr_disks;         /*  9 total disks in the raid set             */
        __u32 raid_disks;       /* 10 disks in a fully functional raid set    */
        __u32 md_minor;         /* 11 preferred MD minor device number        */
        __u32 not_persistent;   /* 12 does it have a persistent superblock    */
        __u32 set_uuid1;        /* 13 Raid set identifier #2                  */
        __u32 set_uuid2;        /* 14 Raid set identifier #3                  */
        __u32 set_uuid3;        /* 14 Raid set identifier #4                  */
        __u32 gstate_creserved[MD_SB_GENERIC_CONSTANT_WORDS - 16];
        /* Generic state information */
        __u32 utime;            /*  0 Superblock update time                  */
        __u32 state;            /*  1 State bits (clean, ...)                 */
        __u32 active_disks;     /*  2 Number of currently active disks        */
        __u32 working_disks;    /*  3 Number of working disks                 */
        __u32 failed_disks;     /*  4 Number of failed disks                  */
        __u32 spare_disks;      /*  5 Number of spare disks                   */
        __u32 sb_csum;          /*  6 checksum of the whole superblock        */
        __u64 events;           /*  7 number of superblock updates (64-bit!)  */
        __u32 gstate_sreserved[MD_SB_GENERIC_STATE_WORDS - 9];
        /* Personality information */
        __u32 layout;           /*  0 the array's physical layout             */
        __u32 chunk_size;       /*  1 chunk size in bytes                     */
        __u32 root_pv;          /*  2 LV root PV */
        __u32 root_block;       /*  3 LV root block */
        __u32 pstate_reserved[MD_SB_PERSONALITY_WORDS - 4];
        /* Disks information */
        mdp_disk_t disks[MD_SB_DISKS];
        /* Reserved */
        __u32 reserved[MD_SB_RESERVED_WORDS];
        /* Active descriptor */
        mdp_disk_t this_disk;
} mdp_super_t;
```

Výpis č. 3: definice RAID superbloku: část hlavičkového souboru md_p.h

popis superbloku, který se nám bude později hodit při řešení některých problémových situacací popsaných níže. Čtenář, který se nechce RAIDem zabývat do hloubky, může tuto část přeskočit.

RAID superblok (o velikosti 4 kB) je uložen na konci diskového oddílu. Jeho pozici získáme buď ze systémového logu (ovladač RAIDu loguje updaty RAID superbloku):

```
md: updating md0 RAID superblock on device
kernel: sdb1 [events: 000000e8](write)\
 sdb1's sb offset: 8956096
```

nebo ji také můžeme odvodit z velikosti diskového oddílu (komentář zdrojového kódu **raidutils**):

```
If x is the real device size in bytes,\
 we return an apparent size of:
y = (x & ~(MD_RESERVED_BYTES - 1))\
 - MD_RESERVED_BYTES
and place the 4kB superblock at offset y.

#define MD_RESERVED_BYTES  (64 * 1024)
```

Nejjednodušší je ale podívat se do `/proc/mdstat`, kde je uvedena velikost každého svazku v blocích, např. u pole typu RAID 1:

```
md0 : active raid1 sdb1[1] sda1[0]\
 131968 blocks [2/2] [UU]
```

RAID superblok z diskového oddílu `sdb1` pak získáme a uložíme do souboru `/tmp/superblok` příkazem:

```
dd if=/dev/sdb1 of=/tmp/superblok bs=1k\
 skip=131968 count=4
```

Jestliže se jedná o pole RAID 0, složené ze dvou disků, pak velikost získanou z `/proc/mdstat` vydělíme 2 např:

```
md4 : active raid0 sdd1[1] sdc1[0]\
 17942272 blocks 16k chunks
```

pak superblok zkopírujeme příkazem:

```
dd if=/dev/sdb1 of=/tmp/superblok bs=1k\
 skip=8971136 count=4
```

Prohlédnout si jej můžeme např. pomocí utility **od** (s vhodnými parametry, např. `od -Ax -tx4`). Pro kontrolu, superblok vždy začíná „magickým číslem“ `0xa92b4efc`. Superblok obsahuje zejména následující informace:

- verzi ovladače raidu, kterým byl vytvořen,
- jedinečný identifikátor pole,
- typ RAIDu,
- datum vytvoření RAID svazku,
- počty disků (aktivních, rezervních apod.),
- preferované vedlejší číslo RAID zařízení,
- stav pole,
- kontrolní součet superbloku,
- počet updatů superbloku,
- datum posledního updatu superbloku,
- velikost stripu,
- informace o stavu jednotlivých diskových oddílů.

Perzistentní superbloky a RAID 0 / linear

Pokud provozujeme pole RAID 0 či linear, máme možnost zvolit variantu bez použití perzistentního superbloku. Volba „`persistent-superblock 0`“, znamená, že se RAID superblok nebude ukládat na disk. Tato možnost existuje z důvodů zachování kompatibility s polemi zřízenými pomocí starší verzí ovladače RAIDu. Po vypnutí takového pole nezůstane na svazku informace o konfiguraci a stavu pole. Proto je tato pole nutné vždy znovu inicializovat při každém startu buď utilitou `mkraid`, nebo pomocí utility `raid0run` (což je pouze symbolický odkaz na `mkraid`) a nelze využít automatického startování polí jádrem při bootu.

Poznámka: Na tuto volbu je třeba dávat pozor při konfiguraci — pokud při konfiguraci pole RAID 0 direktivu `persistent-superblock` vynecháme, použije se standardní hodnota 0, tedy pole bez perzistentních superbloků!

Monitorování stavu pole

Aktuální stav diskových polí zjistíme vypsáním souboru `/proc/mdstat`. První řádek obsahuje typy polí, které ovladač podporuje (záleží na konfiguraci jádra). U jednotlivých RAID svazků je pak uvedeno které diskové oddíly svazek obsahuje, velikost svazku, u redundantních polí pak celkový počet konfigurovaných oddílů a z toho počet funkčních, následovaný schématem funkčnosti v hranatých závorkách. Následující příklad uvádí stav funkčního pole RAID 1:

```
md0 : active raid1 hdc1[1] hda1[0]\
 136448 blocks [2/2] [UU]
```

Druhý příklad uvádí stav pole RAID 1 po výpadku jednoho disku, oddíl `sdc1` je označen jako nefunkční (F=Failed místo čísla aktivního oddílu):

```
md0 : active raid1 sdc1[F] sdd1[0]\
 8956096 blocks [2/1] [U_]
```

Třetí příklad ukazuje stav pole RAID 1, kdy probíhá rekonstrukce:

```
md1 : active raid1 hdc2[1] hda2[0] 530048\
 blocks [2/2] [UU] resync=4% finish=6.7min
```

Součástí **raidtools** bohužel není utilita k monitorování stavu diskových polí, takže si administrátor musí vypomoci skriptem, který je pravidelně spouštěn z **cronu** a kontroluje `/proc/mdstat` (jednoduše např. tak, že si skript na disk uloží obsah `/proc/mdstat` nebo jeho MD5 součet a následně kontroluje, jestli se `/proc/mdstat` změnil; v případě změny pak prostřednictvím e-mailu uvědomí administrátora) anebo filtrem systémového logu.

Rekonstrukce pole

Redundantní typy polí je třeba po inicializaci, po výměně disku, nebo po nahrazení vadného disku rezervním (viz direktiva `spare-disk` v `/etc/raidtab`) rekonstruovat či synchronizovat. Ve všech případech systém rekonstrukci spouští automaticky. Průběh rekonstrukce je možné sledovat v `/proc/mdstat` (viz příklad o několik řádek výše). Rekonstrukce probíhá s nízkou prioritou, nezabere tedy čas procesoru na úkor jiných aplikací, ale bude se snažit využít maximální prostupnosti I/O zařízení. Proto můžeme po dobu rekonstrukce pozorovat zpomalení diskových operací. Maximální rychlost rekonstrukce ovšem také lze ovlivnit

nastavením limitu v /proc/sys/dev/md/speed-limit, výchozí hodnota je 100 kB/sec. Ovladač RAIDu umí současně spustit rekonstrukci na několika polích současně. Pokud jsou však oddíly jednoho disku součástí více polí, které by se měly synchronizovat současně, provede se synchronizace polí postupně (z důvodu výkonu). V systémovém logu se pak objeví neškodné hlášení typu „XX has overlapping physical units with YY“:

```
md: syncing RAID array md1
md: minimum _guaranteed_ reconstruction\
 speed: 100 KB/sec.
md: using maximum available idle IO\
 bandwith for reconstruction.
md: using 128k window.
md: serializing resync, md2 has overlapping\
 physical units with md1!
md: md1: sync done.
md: syncing RAID array md2
md: minimum _guaranteed_ reconstruction\
 speed: 100 KB/sec.
md: using maximum available idle IO bandwith\
 for reconstruction.
md: using 128k window.
md: md2: sync done.
```

V /proc/mdstat jsou ty svazky, na kterých je rekonstrukce pozastavena, označeny jako plně funkční, ale je u nich poznámka resync=DELAYED:

```
Personalities : [linear] [raid0] [raid1] [raid5]
read_ahead 1024 sectors
md2 : active raid1 hdc3[1] hda3[0] 530048\
 blocks [2/2] [UU] resync=DELAYED
md1 : active raid1 hdc2[1] hda2[0] 530048\
 blocks [2/2] [UU] resync=4% finish=6.7min
md0 : active raid1 hdc1[1] hda1[0] 136448\
 blocks [2/2] [UU]
```

Pokud zformátujeme a připojíme čerstvě zřízené redundantní pole, na kterém probíhá rekonstrukce, mohou se v systémovém logu objevit následující neškodné hlášení (je to způsobené tím, že mke2fs, fsck a ovladač FS používají při přístupu jinou velikost bloku, než je výchozí velikost se kterou pracuje ovladač raidu):

```
kernel: set_blocksize: b_count 1, dev md(9,3),\
 block 96765, from c014
kernel: set_blocksize: b_count 1, dev md(9,3),\
 block 96766, from c014
kernel: set_blocksize: b_count 2, dev md(9,3),\
 block 96767, from c014
kernel: md3: blocksize changed during write
kernel: nr_blocks changed to 32 (blocksize 4096,\
 j 24160, max_blocks 385536)
```

Redundantní pole: výměna disku, hot plug

Pokud při čtení nebo zápisu na některý z diskových oddílů, který je součástí redundantního diskového pole, dojde k chybě, je dotyčný oddíl označen jako vadný a pole jej přestane používat. Pokud máme v daném diskovém poli zařazen jeden nebo více rezervních disků (direktiva spare-disk), je tento v případě výpadku automaticky aktivován, systém provede rekonstrukci pole a průběh rekonstrukce zaznamená do systémového logu. V opačném případě pole zůstane v provozu v degradovaném režimu, pak to v systémovém logu bude vypadat zhruba takto:

```
kernel: SCSI disk error : host 0 channel 0\
 id 4 lun 0 return code = 28000002
kernel: [valid=0] Info fld=0x0, Current sd08:11:\
 sense key Hardware Error
kernel: Additional sense indicates\
 Internal target failure
kernel: scsidisk I/O error:\
 dev 08:11, sector 2625928
kernel: raid1: Disk failure on sdb1,\
 disabling device.
kernel:        Operation continuing\
 on 1 devices
kernel: md: recovery thread got woken up ...
kernel: md0: no spare disk to reconstruct\
 array! --- continuing in degraded mode
kernel: md: recovery thread finished ...
```

Příjemnou vlastností diskových polí je také možnost výměny disku za chodu systému. Samozřejmě k tomu potřebujeme v prvé řadě hardware, který to umožňuje. Ovladače slušných SCSI řadičů umožňují přidávání či ubírání zařízení, to ale samo o sobě nestačí. Je zapotřebí používat **SCA** disky určené pro **„hot swap“** a odpovídající SCSI subsystém s SCA konektory a elektronikou, která zajistí stabilitu SCSI sběrnice při odebírání či přidávání zařízení.

Mějme pole typu RAID 1, ve kterém došlo k chybě na oddílu sdc1. Disk sdc1 je připojen ke kanálu 0 SCSI řadiče 0 a má ID rovno 4:

```
md0 : active raid1 sdc1[F] sdd1[0]\
 8956096 blocks [2/1] [U_]
```

Jak tedy probíhá výměna vadného disku, máme-li k tomu potřebné hardwarové vybavení:

- provedeme raidhotremove /dev/md0 /dev/sdc1, což vyřadí vadný oddíl z pole md0,
- provedeme echo "scsi remove-single-device 0 0 4 0" >/proc/scsi/scsi, ovladač SCSI řadiče „zapomene“ na zařízení na řadiči 0, kanálu 0, ID 4, LUN 0,
- vyjmeme vadný disk sdc,
- vložíme nový disk sdc,
- vykonáme echo "scsi add-single-device 0 0 4 0" >/proc/scsi/scsi, což nový disk zpřístupní systému,
- pomocí utility fdisk vytvoříme diskové oddíly,
- vykonáme raidhotadd /dev/md0 /dev/sdc1, čímž přidáme oddíl sdc1 nového disku do pole md0 a na pozadí se spustí rekonstrukce pole.

Pokud nemáme hardware potřebný k „hot-swap“ výměně disků, musíme se smířit s vypnutím systému, výměnou vadného disku a opětovným zapnutím systému. Potom stačí pouze vytvořit pomocí fdisku odpovídající diskové oddíly a příkazem raidhotadd je zařadit do diskového pole. Příkazy pro přidávaní a ubírání SCSI zařízení jsou popsány ve zdrojovém kódu jádra (soubor linux/drivers/scsi/scsi.c).

IDE nebo SCSI?

Doposud jsme se zbývali obecně fungováním softwarového RAIDu, ovšem ve chvíli kdy se rozhodneme sestavit systém s RAIDem, musíme se zamyslet nad výhodami a nevýhodami těchto rozhraní, které s provozováním diskových polí souvisejí. Protože tato problematika je sama o sobě rozsáhlá, uvedeme na tomto místě pouze několik zásadních rozdílů:

Rozhraní SCSI má několik rysů, které napomáhají vyššímu výkonu diskového subsystému:

- **Tagged command queing**: SCSI zařízení mohou přijmout současně více příkazů (běžné SCSI řadiče až 256, disky obvykle až 64) a postupně je zpracovat v libovolném pořadí (možnost optimalizace). Toto je výhodou zejména u zařízení s dostatečnou cache (dnešní disky mají běžně 2 MB, případně 4 MB i více).
- **Elevator sorting**: pořadí příkazů je přerovnáno takovým způsobem, aby se ztrácelo co nejméně času přesouváním hlaviček disku.
- Podpora **Disconnect/Reconnect**: SCSI zařízení mohou uvolnit sběrnici pro komunikaci dalším zařízením, po dobu kdy např. zpracovávají příkazy.
- počet zařízení, který lze k SCSI řadičům připojit je poměrně vysoký (běžně 7 či 15 zařízení na jeden kanál, ovšem musíme také myslet na prostupnost).

Rozhraní IDE má naopak několik nedostatků, které komplikují jeho nasazení, zejména:

- na jednom kanálu mohou být připojena pouze 2 zařízení,
- maximální délka kabelu je mnohem kratší než u zařízení SCSI,
- IDE disky zpravidla nepodporují Disconnect/Reconnect, tedy po dobu provádění příkazu drží sběrnici, což způsobuje zbytečné prodlevy, pokud jsou na jednom kanálu oba disky. Jedno zařízení funguje jako tzv. „master“ a druhé jako „slave“. Závada zařízení které je připojené jako „master“ může často znepřístupnit i druhé zařízení, které je připojeno jako „slave“.

Je tedy jasné, že ačkoliv maximální teoretické propustnosti obou rozhraní jsou dnes poměrně vysoké, v prostředí, kde je kladen důraz na reálnou vysokou propustnost nejen při sekvenčním čtení nebo zápisu, je rozhraní SCSI stále volbou číslo jedna. Pokud se rozhodneme budovat systém na bázi IDE, rozhodně se vyplatí obsazovat každý kanál IDE pouze jedním zařízením. (A to jak z důvodu výkonu, tak stability, protože pokud bychom např. sestavili pole RAID 5 z IDE disků a disky by byly na kanálech po dvou, riskujeme v případě výpadku některého z „master“ disků havárii celého pole, protože tím mohou v krajním případě vypadnout disky oba — jak „master“ tak i „slave““.)

Testování

Když zprovozníme redundantní diskové pole, bude nás zajímat i způsob, jakým otestovat jeho odolnost proti výpadku disku. Můžeme k tomu použít utilitu `raidsetfaulty`, která simuluje výpadek disku a označí jej jako vadný (je potřeba mít dostatečně novou verzi balíčku **raidtools**, ve starších verzích tatu utilita chybí). Potom můžeme disk vyřadit z pole příkazem `raidhotremove`, opět přidat příkazem `raidhotadd` a sledovat průběh rekonstrukce pole. Metodu testování tím, že za chodu vytáhneme konektor disku, rozhodně nelze doporučit, protože tímto způsobem můžeme hardware vážně poškodit.

Tipy

Nyní už máme za sebou jak teorii fungování RAIDu, tak z velké části i praxi softwarového RAIDu pod Linuxem. V této poslední části se zaměříme na méně obvyklé postupy a tipy, jak řešit některé problémové situace.

Jak založit redundantní pole v degradovaném režimu

Softwarový RAID je poměrně flexibilní. Pokud jsme například v situaci, kdy potřebujeme převést systém běžící na samostatném disku na redundantní pole RAID 1, nemusíme kvůli tomu reinstalovat systém. Můžeme využít toho, že lze vytvořit pole v degradovaném režimu. Dejme tomu, že máme instalovaný systém na diskovém oddílu `sda1` a pro zjednodušení je to jediný oddíl na disku `sda`. Do počítače jsme přidali disk stejně velký disk `sdb` a máme připraveno jádro podporující sofwarový RAID. Pomocí `fdisku` vytvoříme diskový oddíl `sdb1` obdobně jako je na disku `sda`. Nyní vytvoříme konfigurační soubor `/etc/raidtab`:

```
raiddev /dev/md0
raid-level                1
persistent-superblock     1
nr-raid-disks             2
nr-spare-disks            0
device                    /dev/sdb1
raid-disk                 0
device                    /dev/sda1
failed-disk               1
```

Pomocí `mkraid` inicializujeme pole, vytvoříme na něm souborový systém (např. pomocí `mke2fs`), připojíme a zkopírujeme na něj data z `sda1`. Odpovídajícím způsobem upravíme konfigurační soubory na svazku `md0` (`/etc/fstab` apod.). Po té systém nabootujeme ze svazku `md0` (třeba pomocí diskety anebo upravíme konfiguraci pro **LILO**), zkontrolujeme, že je vše v pořádku a příkazem `raidhotadd` přidáme oddíl `sda1` do pole.

Jak přidat třetí aktivní disk do pole RAID 1

Pokud máme pole RAID 1 tvořené dvěma disky a rozhodneme se pro zvýšení redundance přidat ještě třetí, nestačí na to pouze příkaz `raidhotadd`. Ten totiž disk do pole přidá, ale pouze jako disk rezervní („spare“). Pokud chceme, aby byl třetí disk také aktivní, musíme si opět pomoci direktivou `failed-disk` v konfiguračním souboru `/etc/raidtab`. Třetí disk označíme jako `failed-disk` a nezapomeneme zvýšit celkový počet aktivních disků na 3 (`nr-raid-disk`):

```
raiddev /dev/md0
raid-level                1
persistent-superblock     1
nr-raid-disks             3
nr-spare-disks            0
device                    /dev/sda1
raid-disk                 0
device                    /dev/sdb1
raidd-disk
```


```
device                  /dev/sdc1
failed-disk
```

Poté pomocí `mkraid` pole znovu inicializujeme a příkazem `raidhotadd` přidáme oddíl `sdc1`. (Tím, že označíme nový disk jako `failed-disk` zajistíme, že jej `mkraid` při inicializaci přeskočí, ale pole bude počítat se 3 disky, následné spuštění `raidhotadd` pak zajistí aktivaci.)

Když z RAIDu 5 vypadne více disků

RAID 5 je odolný vůči výpadku jednoho disku. Protože se stripuje, nejsou data z jednotlivých disků samostatně použitelná a tato situace je téměř neřešitelná. Co tedy dělat v případě, když k výpadku více než jednoho disku dojde? Pokud disky zůstaly po výpadku pole alespoň částečně použitelné, můžeme se pokusit o obnovení pole následujícím způsobem, opět s využitím direktivy `failed-disk`. Utilita `mkraid` v podstatě pouze zapíše na oddíly RAID svazku superbloky a nastartuje pole — nijak tedy nemění obsah oddílů. Teprve jaderný ovladač RAIDu spouští rekonstrukci a tomu můžeme zabránit tím, že pomocí `mkraid` pole znovu inicializujeme, ale pouze v degradovaném režimu (upravíme `raidtab`). Pole pak můžeme připojit s příznakem pouze pro čtení a zjistit, nakolik jsou data na svazku použitelná. Toto můžeme podle potřeby opakovat a postupně z pole vynechat jiný z disků, které z pole v době havárie vypadly, až najdeme takovou kombinaci, při které je souborový systém poškozen nejméně (můžeme zkusit spustit `fsck`). Potom můžeme svazek připojit i pro zápis a pomocí `fsck` souborový systém naostro opravit. Na závěr můžeme přidat i poslední chybějící disk pomocí `raidhotadd`, což vede ke spuštění rekonstrukce pole. Tato metoda ale představuje krajní řešení a rozhodně od ní nelze očekávat zázraky.

Když se změní pořadí či jména disků

Pokud se změní pořadí nebo jména zařízení (např. přidání dalších disků či periferií), které tvoří RAID svazek a nepoužíváme automatické startování RAIDu jádrem při bootu, musíme odpovídajícím způsobem upravit konfigurační soubor `/etc/raidtab` a případně pole znovu inicializovat pomocí `mkraid -force` (pozor – pouze pro ty, kteří vědí, co dělají). Pokud používáme vlastnost „raid autodetect“ jádra, ovladač RAIDu si poradí sám a pole sestaví a spustí podle informací uložených v RAID superblocích diskových oddílů.

Rozšíření pole, konverze raidu na jiný typ

Utilita **raidreconf**, kterou původně vyvíjel Jakob Oestergaard (7), nyní vyvíjená jako open source aktivita Connexem (8), umí zmenšovat, zvětšovat RAID 0 a 5 svazky, převádět svazky typu RAID 0 na RAID 5, přidat nový disk do RAIDu 1 a 5, vytvořit pole RAID 0 ze samostatných disků. Tato utilita ale není dostatečně testována, takže pozorné čtení manuálu a záloha je naprostou nutností! Spolu s utilitou `resize2fs` je tedy možné tedy možné měnit i velikost polí.

Boot raid

Je možné provozovat kořenový svazek na RAIDu, a to typu linear, RAID 0 a RAID 1. Konfigurace starších verzí LILa byla sice trochu problematická, ale nové verze LILa již RAID podporují přímo (9), (10).

Jak potlačit autodetekci RAID polí

Pokud máme jádro s podporou autodetekce RAID polí a z nějakého důvodu potřebujeme autodetekci dočasně vypnout, můžeme jádru při startu zadat parametr `raid=noautodetect`.

RAID a swap

V souvislosti s RAIDem se často diskutuje o tom, zda má smysl vytvářet odkládací oddíly na RAIDu a když ano, tak jaký typ pole použít. Tady je zapotřebí vzít v potaz omezení daná ovladačem RAIDu (viz výše omezení platná pro jádra 2.2.x) a dále se musíme rozhodnout, zda nám jde o zrychlení swapování, nebo o robustnost systému. Pokud nám jde o rychlost, můžeme jako odkládací oddíl použít svazek RAID 0; ovšem podobného efektu můžeme dosáhnout i bez použití RAIDu a to tak, že v `/etc/fstab` uvedeme u odkládacích oddílů stejnou prioritu:

```
/dev/sda3      none    swap    sw,pri=1
/dev/sdb3      none    swap    sw,pri=1
```

Používat pro swap RAID 0 je tedy v podstatě zbytečné. Naopak v případě, že nám jde o robustnost systému, můžeme pro odkládací oddíl s výhodou použít RAID 1 svazek. Systém pak s výpadkem disku nepřijde o část swapu.

Výkon a stabilita

Nejprve srovnejme výkon softwarového raidu pod jádry 2.2 a 2.4: RAID 0 je rychlejší u jader 2.4, RAID 1 je na tom zhruba stejně, RAID 5 byl na řadě 2.4 z počátku výrazně pomalejší, i když toto se v poslední době rychle mění a nyní je výkon srovnatelný nebo lepší. Pokud jde o srovnání rychlosti softwarového RAIDu a hardwarových řešení, softwarový RAID je oproti hardwarové implementaci samozřejmě náročnější na systémové prostředky, ale na druhou stranu bývá mnohdy rychlejší (výrazně rychlejší bývá zejména RAID 0). (Poznámka: pro vylepšení výkonu RAIDu 1 při čtení existuje záplata ovladače RAIDu „readbalance“.)

Pokud jde o robustnost implementace, stabilita RAIDu typů linear, RAID 0 a 1 je poměrně vysoká, naopak nasazení RAIDu 5 v ostrém provozu ještě nelze doporučit. V této souvislosti ještě zmíním jednu vlastnost Linuxové implementace softwarového RAIDu: V případě jakékoliv I/O chyby ovladač RAIDu okamžitě daný diskový oddíl z RAIDu vyřadí, bez ohledu na to, jestli se jedná o chybu fatální, anebo o případ, kdy by třeba stačilo danou I/O operaci zopakovat. Jinými slovy disk, který občas vrátí nějakou chybu, ale je nadále více méně schopný fungovat (a který by systém nadále používal, pokud by nebyl součástí RAID svazku, ale byl připojený jako samostatný oddíl), linuxový ovladač přestane používat. Tím se zbytečně snižuje robustnost RAIDu, protože snadněji může dojít k situaci, kdy z pole vypadne postupně i více disků, než kolik je k provozu daného pole třeba a pole zhavaruje. Proto lze doporučit použití rezervních disků a vyhnout se shánění rezervního disku na poslední chvíli, kdy už pole mezitím běží v degradovaném režimu. Ze srovnání softwarových RAID implementací Linuxu, Windows 2000 a Solarisu (11) vyplývá, že linuxový RAID ve výchozím nastavení provádí rekonstrukci se sníženou prioritou a limi-

tovanou rychlostí, takže probíhající rekonstrukce mnohem méně negativně ovlivňuje výkon systému po dobu rekonstrukce. (Poznámka: V odkazovaném srovnání ovšem autoři opakovaně chybně uvádějí absenci některých vlastností linuxové softwarové implementace RAIDu.)

Závěrem

Softwarový RAID je cenově lákavou alternativou nákladných hardwarových řešení. Další výhodou je flexibilita (např. možnost sestavení pole v degradovaném režimu, možnost eventuální částečné záchrany dat v případě výpadku celého pole, protože je známá struktura dat v diskovém poli, konverze RAID svazků z jednoho typu RAIDu na jiný). Některé z těchto možností jsou ale spíše experimentálního rázu. Za spolehlivé lze označit implementace RAIDu typu linear, RAID 0 nebo RAID 1. Softwarový RAID je náročnější na systémové prostředky než hardwarová řešení, některé typy (zejména RAID 0) ovšem mohou být výrazně rychlejší než hardwarové varianty. Je tedy na adminstrátorovi, aby zvážil výhody a nevýhody sofwarového či hardwarového RAIDu vzhledem k aktuálním podmínkám.

Tento článek ani v nejmenším nenahrazuje dokumentaci k ovladači Linuxového softwarového RAIDu či obslužným utilitám — proto zde až na výjimky záměrně nejsou komentovány přepínače obslužných utilit. Důkladné čtení dokumentace (nebo v případě nejasností studium zdrojového kódu — dokumentace bohužel stále není úplná) by mělo být samozřejmostí, rovněž existuje konference **linux-raid** s prohledávatelným archívem (12). A ještě úplně poslední poznámka na závěr: nezapomínejme, že (redundantní) RAID chrání pouze před výpadkem určitého počtu disků a rozhodně nenahrazuje nutnost pravidelného zálohování dat. ■

1 ovladač RAIDu pro jádra 2.2.x
ftp://ftp.linux.cz/pub/linux/kernel/people/mingo/raid-patches/
2 raidtools 0.42 (staré)
ftp://ftp.kernel.org/pub/linux/daemons/raid
3 raidtools 0.90 (nové)
http://people.redhat.com/mingo/raid-patches/
4 HOWTO aktuální verze ovladače RAIDu
http://www.linux.cz/linuxdoc/HOWTO/Software-RAID-HOWTO.html
5 HOWTO starší verze ovladače RAIDu
http://www.linux.cz/linuxdoc/HOWTO/Software-RAID-0.4x-HOWTO.html
6 mdctl — nástupce raidtools
http://www.cse.unsw.edu.au/~neilb/source/mdctl/
7 raidreconf — původní verze
http://unthought.net/Software-RAID.HOWTO/
8 raidreconf — nová verze
http://ops.connex.com/Projects.shtml
9 Root-RAID-HOWTO (ke starší verzi RAIDu)
http://www.linux.cz/linuxdoc/HOWTO/Software-RAID-0.4x-HOWTO.html
10 Boot+Root+Raid+LILO HOWTO (k nové verzi RAIDu)
http://www.linux.cz/linuxdoc/HOWTO/Boot+Root+Raid+LILO.html nový root+raid
11 srovnání implementací SW RAIDu
http://www.cs.berkeley.edu/~abrown/papers/usenix00/paper.html
12 archív konference linux-raid
http://marc.theaimsgroup.com/?l=linux-raid

WIN95 + RH6.2 = 10GB HDD

Vítězslav Dvořák, 27. srpna 2001

V počítači jednoho mého přítele byly původně Windows ME, ale majitel neunesl výčitky svědomí a požádal mne, abych mu nainstaloval Linux a jeho legální kopii Microsoft Windows 95 tak, aby mohl pokračovat ve své práci a k tomu ještě mohl mít v počítači plně funkční kopii svých stránek, které má jinak na komerčním webovém serveru.

A teď už k instalaci. Pokud můžete přidělit jednotlivým operačním systémům samostatné disky, máte vyhráno. Bohužel v tomto případě byl k dispozici pouze jeden 10 GB disk.

Potíž je v tom, že Windows 95 (a MS-DOS) neumí FAT32 (pozn. redakce: Windows 95 OSR 2 již 32-bitovou FAT umí). Velkým omezením je zde fakt, že FAT16 může být maximálně 2 GB velká. Pokud tedy hodláte instalovat Windows 98 anebo novější variantu Microsoft Windows, nebudete pravděpodobně mít s instalací na diskový oddíl (partition) „téměř“ libovolné velikosti problém.

Na instalačním CD Red Hatu je dosovský nástroj **fips**, který umí zmenšit diskový oddíl, pokud jsou na jeho konci prázdné bloky, jenomže novější Windows záměrně ukládají na konec disku data, která nejdou normálním způsobem přesunout.

Dokumentace programu `fips` radí použít **SpeedDisk** nebo **defrag** či něco podobného, jenže verze pro FAT32 už bloky z konce disku nepřesunují (!). Z části může jít o záměr, ale z části je to způsob jak zrychlit diskovou odezvu. Konkrétně `SpeedDisk` fyzicky přesouvá data na to místo na disku, kde bude z hlediska systému jeho načítání nejméně zdržovat. To znamená, že po boot recordu jsou to adresáře, za nimi odkládací soubor (swap) a po něm již následují často využívaná data („high access“), která se fyzicky ukládají na střední cylindry disku, kde jsou rozdíly ve vystavovací době kmitající hlavičky pevného disku nejmenší. A konečně na bezprostřední konec prostoru diskového oddílu se ukládají mě nepochopitelná („low access“) data. Bohužel mě dostupný `Norton SpeedDisk` již neměl nástroj `Walk Map`, který umožňuje zobrazit soubory na tom kterém místě disku. A zde se dostáváme k problému: Konec diskového oddílu je obsazen a není tudíž možné jej pomocí nástroje `fips` zmenšit a uvolnit tak prostor pro Linux. Tudíž nezbylo než se kompletně vzdát dědictví minulosti a Windows s kompletním obsahem disku nainstalovat znovu.

1) Instalace Linuxu

Předpokládejme že máme prázdný disk. Při instalaci RH je potřeba na obrazovce, kde se vybírá způsob instalace (možnosti „Pracovní stanice Gnome“, „Pracovní stanice KDE“, „Server“ atd.) zaškrtnout vpravo nahoře přepínač „Použít fdisk“.

Když se při instalaci spustí textové prostředí programu fdisk, vložíme tyto příkazy:

```
n       (vytvoří nový diskový oddíl)
p       (bude to primární oddíl)
3       (bude to 3. primární oddíl ze 4 možných)
+4000M  (velikost oddílu, požadujeme 4GB)
```

Tímto se linuxový oddíl vytvoří tak, aby zbylo místo pro DOS/WIN. Nyní je potřeba pro Linux ještě vytvořit odkládací diskový oddíl („swap partition“). Odkládací oddíl by měl mít minimálně dvojnásobek velikosti operační paměti RAM (pozn. redakce: toto pravidlo platí zejména pro linuxová

jádra 2.4.x, která se objevují v novějších distribucích — např. Red Hat 7.x. Distribuce Red Hat 6.2 obsahuje jádro řady 2.2, kde za předpokladu, že je dostatek fyzické paměti, postačí na odkládací oddíl i polovina velikosti fyzické paměti.):

```
n     (diskový oddíl vytvoříme obvyklým způsobem)
p     (primární - jiné nastavení doporučuji
      opravdu jen zkušeným uživatelům)
4     (využijeme místo na konci disku)
+128M (velikost oddílu pro swap)
```

Vytvořili jsme nový diskový oddíl, avšak ten je zatím systémem považován za standardní linuxový oddíl („Linux Native“). Nyní je potřeba systému říci, aby tento oddíl považoval za odkládací:

```
t       (příkaz pro změnu typu diskového oddílu)
4       (měníme nastavení čtvrtého oddílu)
82      (číslo typu "Linux swap")
```

Nyní jsou náležitosti nutné pro Linux hotovy, zapíšeme tedy novou tabulku rozdělení na disk a současně ukončíme fdisk stiskem „w“.

Instalační program Red Hat Linuxu nám nyní zobrazí nástroj **Disk Druid**. Vybereme položku náležící k oddílu hda3. Po stisknutí tlačítka upravit vložíme jako název místa připojení lomítko „/“ čímž ho ustanovíme nadobro linuxovým diskem. (Odkládací oddíl (swap) není třeba nijak nastavovat.)

Po potvrzení začne samotná instalace systému. V dialogu „kam uložit zavaděč systému“ (**LILO**) potvrdíme předvolenou hodnotu „MBR“. Poté se nainstaluje systém Linux. Při instalaci je důležité nechat vytvořit startovací disketu systému Linux. Pak nastává druhá fáze:

2) Instalace MS Windows 95

Nejdříve je potřeba z diskety nebo CD nabootovat do DOSu (příkazového řádku Microsoft Windows 95) a spustit dosovský program FDISK ve kterém zadáme tyto příkazy:

```
1       (Vytvoř nový diskový oddíl)
1       (ano, chceme aby byl primární)
A/Y     (Ano/Yes - chceme, aby měl maximální
        možnou velikost a spouštěl
        se z něj systém)
```

Nyní již máme připraven 2 GB velký disk C:, kam je možno nainstalovat Windows, ale stále máme na disku 3 GB místa, které je nevyužito. Abychom i tento zbylý prostor zpřístupnili pro Windows, vytvoříme diskový oddíl typu extended, kterýžto v sobě může obsahovat logické diskové oddíly pro další disky D: a E: těmito příkazy:

```
1       (Vytvoř nový diskový oddíl)
2       (nyní potřebujeme rozšířený
        "EXTENDED" oddíl)
A/Y     (Ano/Yes - chceme aby měl
        maximální možnou velikost)
```

A nyní již jen vytvořit logické disky D: a E:

```
1       (Vytvoř nový diskový oddíl)
3       (chceme logický disk v rozšířeném oddílu)
A/Y     (Ano/Yes - chceme aby měl
        maximální možnou velikost)
```

Stejnou sekvenci (1,3,A) zopakujeme ještě jednou, abychom přiřadili disku E: i poslední zbylý 1 GB místa.

Práce s dělením disku je hotova. Stiskem klávesy ESC ukončíte program FDISK a restartujeme počítač, aby se projevily naše změny. Takže disk nakonec vypadá takto:

```
hda1 primary * FAT 16        "C:"      2GB
hda2 extended  FAT           (D: + E:) 3GB)
hda3 primary   LINUX NATIVE  "/"       4GB
hda4 primary   LINUX SWAP              128MB\
                             (pro 64MB RAM)
hda5 logical   FAT           "D:"      2GB
hda6 logical   FAT           "E:"      1GB
```

Po restartu je potřeba nabootovat z diskety nebo CD znovu do DOSu. Jestli je vše v pořádku, si můžeme ověřit zadáním příkazu pro změnu aktuální jednotky disku:

```
C:      (přejdi na disk C:)
```

Po stisku klávesy Enter by se již mělo objevit klasické hlášení příkazové řádky DOSu C:\>. Obdobně můžeme vyzkoušet i existenci ostatních disků D: a E: Avšak z těchto disků není zatím možné číst, ani na ně zapisovat. Na tyto činnosti je připravíme formátováním pomocí příkazu FORMAT:

```
FORMAT C: /S
FORMAT D:
FORMAT E:
```

Na dotazy ohledně ztráty veškerých dat odpovídejte ano, protože tyto disky jsou zatím méně než prázdné a není na nich tudíž co ztratit. Tuto fázi je možné vynechat, protože instalační program Microsoft Windows je schopen si disk před instalací zformátovat sám, eventuálně máte ve složce „Tento počítač“ v místní nabídce (pravé tlačítko myši nad ikonou) každého disku možnost „Naformátovat“.

Pokud jste si však disky zformátovali, již nyní můžete si instalaci podle rychlosti vaší CD-ROM mechaniky výrazně urychlit když si zkopírujete instalační soubory na pevný disk pomocí následujících příkazů. Nejprve vytvoříme na disku nový prázdný adresář:

```
MD c:\win95
```

Zkopírujeme instalační soubory z F: (CD-ROM) příkazem:

```
COPY f:\win95\*.* c:\win95
```

A instalace Windows 95 může začít:

```
C:
CD win95
SETUP
```

Doporučuji také ještě nechat si pomocí příkazu SETUP /? vypsat seznam použitelných parametrů instalace, díky kterým nás nebude instalace zdržovat například zbytečnou kontrolou našich nových disků, nebo instalací síťového software do počítače bez síťové karty. Windows 95 nainstalujeme pak již standardním způsobem.

3) Startovací nabídka pro Linux / Windows

S největší pravděpodobností instalace Microsoft Windows zrušila startovací nabídku programu **LILO** a startují rovnou Windows, nebo počítač při svém startu nabízí pouze možnost zavedení systému Linux. Abychom to vše uvedli do správného stavu potřebujeme změnit nastavení souboru /etc/lilo.conf. Nejjednodušší to bude pomocí ná-

stroje **linuxconf**. Nabootujeme Linux (pokud je to nutné, tak z diskety), přihlásíme se jako uživatel root a příkazem `linuxconf` spustíme konfigurační nástroj Red Hat Linuxu.

V nabídce nalezneme následující dialog „startovací mód/LILO/nastavit konfigurace ostatních OS v LILO“ zde zvolíme volbu „přidej“ a vyplníme dvě v dialogu obsažená políčka:

- jmenovka: název systému zobrazovaný jako možnost při startu počítače. Většinou „dos“, „win“ či „windows“;
- oddíl pro start: tedy ze kterého oddílu se má bootovat (v našem případě je to `/dev/hda1`).

Stiskem tlačítka „akceptuj“ potvrdíme změny v konfiguraci zavaděče LILO a stiskem tlačítka „aktivovat změny“ pak ukončíme `linuxconf` s provedením změn. Setkal jsem se s tím, že v některých případech se však změny neaktivují, proto ještě pro jistotu spustíme LILO ručně příkazem `/sbin/lilo`. Po spuštění LILO vypíše zhruba následující:

```
Added linux
Added dos *
```

Po restartu nám již počítač (po stisku klávesy TAB) nabízí možnost výběru systému.

4) Zpřístupnění disků C:, D:, E: pro Linux

Nejprve je potřeba vytvořit adresáře, do kterých se budou jednotlivé disky připojovat:

```
mkdir /mnt/c
mkdir /mnt/d
mkdir /mnt/e
```

Nyní ještě zbývá počítači říci, aby tyto disky připojil již při startu Linuxu – což je určeno v souboru `/etc/fstab`, do kterého přidáme následující řádky:

```
/dev/hda1   /mnt/c   vfat   defaults   0 0
/dev/hda5   /mnt/d   vfat   defaults   0 0
/dev/hda6   /mnt/e   vfat   defaults   0 0
```

Po uložení již můžeme disky připojit příkazem `mount`:

```
mount /mnt/c
mount /mnt/d
mount /mnt/e
```

A to je vše, nyní máme funkční instalaci Red Hat Linuxu 6.2 a Microsoft Windows 95. ■

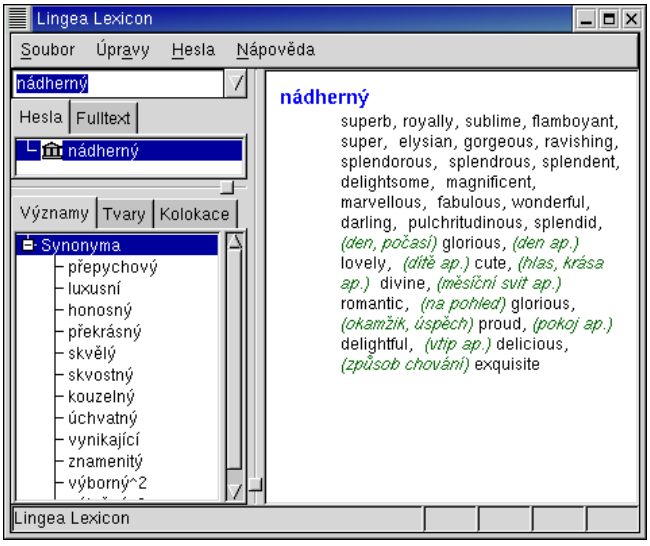
Lingea Lexicon 2000 pro Linux

David Häring, 4. července 2001

Slovník je věc veskrze užitečná — konec konců nějaký ten slovník – minimálně v knižní podobě — by doma našel asi každý. Nějaký ten pátek už ovšem klasickým „papírovým“ slovníkům konkurují slovníky elektronické – zejména lidé používající hojně cizí jazyky si už nedovedou představit moderní desktop bez rozsáhlého slovníku. Stejně tak jsou populární aplikace usnadňující výuku cizích jazyků. Uživatelé linuxových systémů ovšem v minulosti příliš na výběr neměli. S výjimkou stáhnutelné verze slovníku **Xdict**, který již nyní zřejmě není dostupný (1), prakticky žádný česko-anglický / anglicko-český slovník neexistoval. Jedinou možností zůstávaly „online“ slovníky (namátkou uveďme např. (2), (3), (4), (5), (6), (7), (8)) umístěné na Internetu, ovšem tyto slovníky vzhledem k pomalosti a ceně přístupu (pokud dotyčný neměl přístup k pevné lince) představují pouze nouzové řešení. V této pro uživatele Linuxu nemilé situaci nastal obrat ve chvíli, kdy se firma Lingea (9) rozhodla portovat své slovníky Lingea Lexicon na platformu Linux. Jako první byly na trh uvedeny Velký anglicko-český a česko-anglický slovník a Velký německo-český a česko-německý slovník.

Lingea Lexicon — velký anglicko-český / česko-anglický slovník

Tento slovník obsahuje 170000 hesel (250000 významů). K dispozici je také varianta „studijní“, která se od velkého slovníku liší menším rozsahem (75000 hesel) a absencí fulltextového vyhledávání. Jinak slovník k zadanému heslu umí vyhledat slova nějakým způsobem příbuzná: synonyma, antonyma, slova s příponami, frázová slovesa či slovní spojení. Slovník také obsahuje výukové prvky — je např. možné si prohlížet a přitom poslouchat výslovnost slovíček sdružených do tématických okruhů a následně se nechat přezkoušet.



Lingea Lexicon v akci

Linuxová verze slovníku má zatím oproti verzi pro Microsoft Windows některá omezení, které by měly být v dalších verzích postupně odbourány, např. funkce „vždy navrchu“ nebo funkce „automatické přebírání schránky“. Také absence nápovědy nepotěší — k dispozici je pouze dokumentace v „papírové“ podobě.

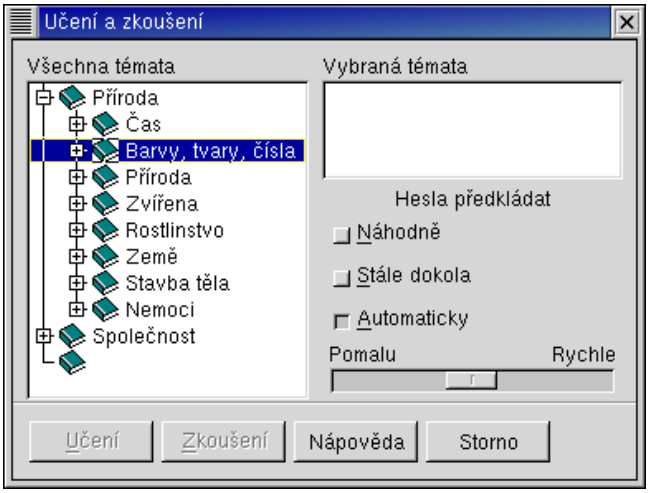
Dostupnost

První verze slovníku bezproblémově fungovala jen na distribucích postavených na glibc verze 2.2 (např. Red Hat 7.x), neběžela tedy např. na Red Hatu 6.x (ačkoliv případné nároky na verze software nebyly v dokumentaci nikde uvedeny). Také jsem zaznamenal problémy v prostředí Gnome, pokud si uživatel vybral jiné než výchozí nastavení dekorace oken (témat). Dále chyběl font používaný pro fonetický přepis výslovnosti, což mohl být problém zejména pro uživatele, kteří nemají počítač vybavený zvukovou kartou a nemohou si poslechnout výslovnost slovíček namluvenou rodilým mluvčím. Lingea ovšem na problémech zapracovala, takže aktuální

verze slovníku už výše popisovanými problémy (s výjimkou absence nápovědy) netrpí.

Instalace

Je řešena pomocí instalátoru s grafickým rozhraním. Slovník je možné instalovat na lokální disk, anebo jej spouštět přímo z distribučního CD. Instalace je bezproblémová. V případě jednouživatelské instalace se vše instaluje do domovského adresáře uživatele. V případě instalace pro více uživatelů je potřeba, aby instalaci provedl uživatel root, konfigurační soubor je pak uložen v souboru `/etc/lex3.conf`, datové soubory se instalují standardně do `/usr/share/lex3`. V domovském adresáři si při prvním spuštění slovník vytvoří konfigurační soubor `.lex3rc` (tento soubor ale není potřeba editovat ručně, nastavení preferencí je dostupné z menu aplikace) a adresář `.lex3`, který slouží k ukládání slovníků vytvářených uživatelem.



Výběr témat pro výuku

Hodnocení

Slovník je portací již osvědčeného produktu, takže z pohledu návrhu aplikace mu nelze nic vytknout, jedná se o produkt velmi zdařilý. Případná omezení stávající linuxové verze by měla s příchodem dalších verzí odpadnout. Pokud jde o cenu, je stejná jako u verze pro Microsoft Windows (ovšem vlastníci licencí slovníku pro jiný OS zaplatí pouze část této ceny — což je jistě potěší). Závěrem tedy mohu konstatovat, že přes počáteční vady na kráse se jedná o slušný produkt, který českým uživatelům Linuxu dlouho chyběl a nezbývá než doufat, že nezůstane jen u slovníku anglického a německého. ■

1 Xdict
http://www.intersoft.cz/workroom/
2 On-line slovník
http://slovnik.nettown.cz/
3 On-line slovník
http://www.slovnik.cz/bin/ecd
4 On-line slovník
http://melkor.dnp.fmph.uniba.sk/~garabik/slovnik/
5 On-line slovník
http://sweb.cz/freedict/freedict.htm
6 On-line slovník
http://www.dict.org/bin/Dict
7 On-line slovník
http://www.slovnik.cz/bin/ecd
8 On-line slovník
http://www.mtranslations.cz/30/en/dictionary/dictionary.htm
9 Lingea
http://www.lingea.cz

Apcupsd — free obslužný software záložních zdrojů APC

David Häring, 29. července 2001

Proč používat UPS

Každý uživatel, který s počítači pracuje delší dobu, již nepochybně zažil výpadek elektrického proudu. V lepším případě člověk přijde pouze o rozpracovaná data, v horším případě může dojít i k poškození souborového systému a ztracený čas i data pak mohou znamenat nezanedbatelnou finanční ztrátu. Výpadky napájení, ale i jeho kolísání se také v neposlední řadě podepisují na životnosti hardware počítače. Řešení představují záložní napájecí zdroje (UPS, z angl. „Uninterruptible Power Supply“), které jsou schopny v případě výpadku napájení v rozvodné síti po určitou dobu poskytovat napájení z baterií.

Typy záložních zdrojů

Pokud se rozhodneme záložní zdroj pořídit, máme na výběr z několika druhů. Nejjednodušší modely zajišťují pouze základní ochranu před výpadkem napájení — v případě přerušení dodávky el. proudu přepnou na napájení z baterie. Lepší modely současně poskytují ochranu před podpětím či přepětím v rozvodné síti. Dále se jednotlivé typy liší možností komunikace s počítačem: levnější modely pouze signalizují přerušení a obnovení dodávky napájení v síti, „chytřejší modely“ umožňují mimo jiné monitorovat stav baterie — takže je možné přizpůsobit spuštění shutdownu aktuálnímu stavu baterie. Některé UPS komunikují pouze přes sériový port, i když v dnešní době se už pomalu přechází na USB, jiné mohou být navíc spravovány přes síť pomocí SNMP či přes modem.

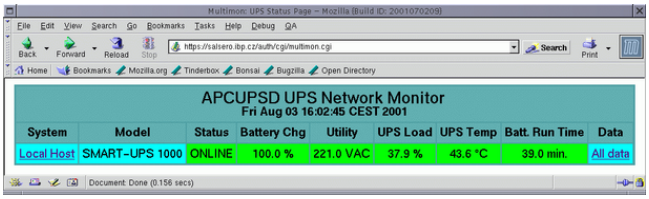
Záložní zdroje a software pro Linux

Většina výrobců záložních zdrojů dnes buď podporuje Linux přímo a dodává obslužný software spolu s UPS nebo jej dává k dispozici prostřednictvím Internetu (příkladem budiž např. APC, Victron, Best Power, Fenton). Řada UPS je pod Linuxem podporována také „free“ softwarem třetích stran (**NUT**, **Apcupsd**). V každém případě je zapotřebí se před zakoupením ujistit, zda je UPS a daný typ komunikačního kabelu pod Linuxem skutečně plně podporován.

Apcupsd a podporované typy UPS

Firma American Power Conversion Corporation (1), zkráceně APC, zpočátku dlouhou dobu nebyla ochotna poskytnout technické informace týkající se popisu komunikačních protokolů svých UPS. Což byla škoda, zejména vzhledem k to-

mu, že záložní zdroje APC byly poměrně rozšířené a i cenově patřily mezi přístupnější kategorie. Proto vznikl projekt **Apcupsd** (2), jehož autoři se rozhodli i za absence technických specifikací vytvořit alternativní obslužný software pro tyto UPS. Apcupsd podporuje všechny běžné modely: např. Back-UPS, Back-UPS Pro, Smart-UPS v/s, Smart-UPS a další. Kabel pro komunikaci s počítačem přes sériový port je zpravidla součástí dodávky UPS. Apcupsd zatím nepodporuje komunikaci přes USB.



Webové rozhraní Apcupsd

Alternativní software

Podívejme se nejprve na software, který poskytuje přímo APC:

APC Simple Signaling Daemon je software pro komunikaci s UPS v základním „simple“ nebo „basic signaling“ režimu pomocí kabelu 940-0020B. Uplatnění nalezne zejména pro ovládání záložních zdrojů typu BackUPS, které komunikují pouze v režimu „basic signaling“. Součástí instalace je kromě vlastního démona `ssd` i skript `apcssd`, který akceptuje volby start/stop a je určen pro instalaci do adresáře `/etc/rc.d/init.d` v distribucích Red Hat. Konfiguruje se pouze sériový port a doba po jejímž uplynutí se spustí shutdown v případě, že nedojde k obnovení dodávky el. proudu. Software je možné získat na serveru APC (3).

APC také vyvíjí obslužný software **PowerChute** a **PowerChute Plus**, který je k dispozici i pro Linux. **PowerChute** je ve srovnání s `apcupsd` zbytečně rozsáhlý a má také výrazně větší nároky na systémové prostředky. To je dáno mimo jiné tím, že na rozdíl od `apcupsd` (který se ovládá z příkazové řádky nebo přes webové rozhraní) poskytuje grafické rozhraní v prostředí X Window System.

PowerChute network shutdown je software pro modely UPS sloužící pro napájení většího počtu počítačů komunikující přes síť pomocí protokolu SNMP.

Pokud jde o software třetích stran, kromě Apcupsd záložní zdroje APC komfortně podporuje také **NUT** (4). Další software jako **genpowerd** (5) anebo **upsd**(6) podporují záložní zdroje APC pouze v omezené míře (nepodporují „smart“ režim a fungují pouze s některými typy kabelů).

Komunikace mezi UPS a počítačem

Modely BackUPS a ShareUPS komunikují s počítačem pomocí již krátce zmíněného „simple signaling“ protokolu. U těchto modelů se komunikace omezuje na několik signálů (výpadek napájení, obnovení napájení, vybití baterie, vypnutí UPS). Modely BackUPS Pro, SmartUPS, MatrixUPS používají tzv. „subsmart signaling“ nebo „smart signaling“ protokol, který umožňuje monitorovat stav baterie, teplotu či vlhkost uvnitř UPS, dotazovat se na nastavení UPS apod. Detailní popis protokolů je součástí dokumentace `apcupsd`.

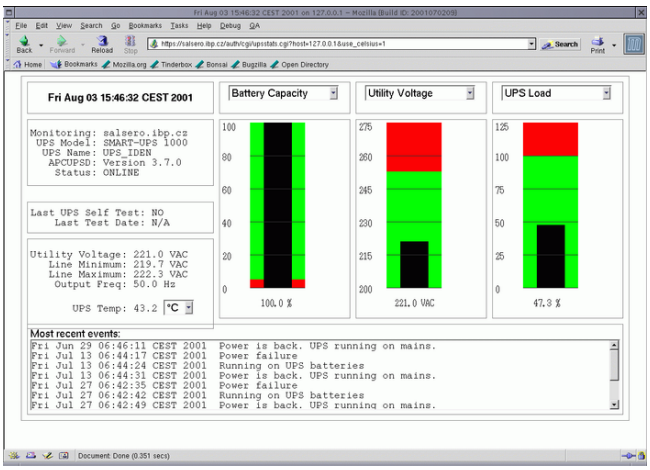
Jak apcupsd funguje?

Apcupsd může běžet buď ve samostatném („standalone“) nebo síťovém („net“) režimu. Pokud máme k UPS připojen pouze jeden počítač, je situace jednoduchá. Apcupsd s UPS komunikuje přes sériový port a v případě potřeby zajistí včasné spuštění shutdownu (a případných dalších akcí definovaných uživatelem). Máme-li silnější záložní zdroj, ke kterému je připojeno napájení více počítačů, pak jeden počítač funguje jako tzv. **master** — monitoruje stav UPS přes sériový port, přes síť pak komunikuje s ostatními počítači a v případě potřeby iniciuje ukončení běhu systému (shutdown) nejen na systému, ke kterému je UPS připojena, ale i na všech ostatních.

Po startu se `apcupsd` nejprve pokusí zjistit, zda je k nakonfigurovaném portu skutečně připojena UPS. Pokud se podaří navázat komunikaci s UPS, spustí další proces, nebo více procesů v závislosti na dané konfiguraci. Pokud je UPS připojena přes sériový port, spustí `apcupsd` proces `apcser` („apc serial“), který obstarává vlastní komunikaci s UPS. Je-li `apcupsd` konfigurován tak, aby sdílel informace o stavu UPS přes síť, spustí `apcser` dále proces `apcnis` („apc network information server“). Pokud je počítač napájen z UPS, která je ovládána z jiného počítače, spustí `apcupsd` místo `apcser` proces `apcslv` („apc slave“). Ten pak komunikuje přes síť s procesem `apcmst` („apc master“), který je spuštěn na počítači, ke kterému je UPS připojena. Proces `apcmst` informuje o stavu UPS a případných výpadcích napájení ostatní podřízené počítače.

Instalace

Apcupsd je dostupný jak formou zdrojových kódů, tak předkompilovaných verzí ve formě RPM instalačních souborů. Pokud chceme instalovat ze zdrojových kódů, je použit **autoconf**, takže kompilace je přímočará (`./configure; make; make install`). Po instalaci je důležité se ujistit, zda-li byly správně instalovány skripty v `/etc/rc.d/init.d` apod.



Webové rozhraní Apcupsd, detailní statistika

Konfigurace

Konfigurace je standardně ukládána v souboru `/etc/apcupsd/apcupsd.conf`. Podívejme se na nejdůležitější konfigurovatelné parametry:

- UPSCABLE — typ sériového kabelu je nutné zadat, zjistíme buď z potisku na kabelu, nebo z popisky produktu.
- UPSTYPE — zadáme typ ups, platná klíčová slova jsou popsána v manuálové stránce apcupsd.
- LOCKFILE — cesta k souboru, který slouží jako zámek, aby se více instancí aplikace nepokoušelo současně přistupovat k sériovému portu.
- NETSERVER — pokud je zapnuto, apcupsd bude poskytovat informace o stavu UPS přes síť.
- FACILITY — apcupsd standardně zapisuje do systémového logu pod kategorií zpráv daemon, pokud chceme mít zprávy apcupsd ve zvláštním souboru, můžeme použít některou z kategorií local0 až local7.
- BATTERYLEVEL — tato direktiva udává minimální nabití baterie (v procentech), pokud nabití baterie klesne pod tuto mez, spustí se shutdown systému.
- MINUTES — chytřejší modely záložních zdrojů umějí odhadnout, na jak dlouhou dobu jsou při výpadku rozvodné sítě ještě schopny zajistit napájení. Pokud tato doba klesne pod mez zadanou parametrem MINUTES, iniciuje UPS shutdown systému.
- TIMEOUT — tato direktiva se na rozdíl od předchozích dvou používá u jednoduchých typů UPS, které průběžně neinformují o stavu baterie. Udává čas v sekundách, po jehož vypršení při výpadku napájení bude spuštěn shutdown systému. Direktivy BATTERYLEVEL, MINUTES a TIMEOUT mohou být nastaveny současně, apcupsd pak inicializuje shutdown při překročení libovolného z těchto tří kritérií.

Direktivy UPSCLASS a UPSMODE slouží k nastavení, zda se jedná o UPS využívanou pouze jedním počítačem, nebo je z jedné UPS napájeno více strojů a apcupsd pak běží v síťovém režimu apod. Se síťovým režimem dále souvisejí direktivy NETTIME, NETPORT, MASTER a SLAVE.

Otestování instalace

Testování instalace je velmi důležité. Abychom se vyhnuli nepříjemnostem při chybném nastavení, je vhodné nejprve UPS připojit k sériovému portu, ale počítač nechat zapojený přímo do rozvodné sítě. Nejdříve se ujistíme, že komunikace s UPS funguje podle našich představ:

- ověříme si výpisem procesů, že apcupsd skutečně běží,
- v systémovém logu by mělo být hlášení o úspěšném startu,
- utilita apcaccess musí vypsat status záložního zdroje,
- pokud přerušíme napájení UPS ze sítě, UPS se musí přepnout na baterie a apcupsd to musí zaregistrovat.

Teprve v případě, že vše funguje tak jak má, můžeme zapojit napájení počítače přes UPS a simulovat výpadek napájení naostro. Zpočátku je vhodné volit kritéria opatrně tak, aby se inicializoval shutdown s rezervou. V opačném případě by UPS vypnula napájení dříve než celá shutdown sekvence stihne proběhnout. Dodatečně pak parametry můžeme upravit.

Sledování stavu UPS

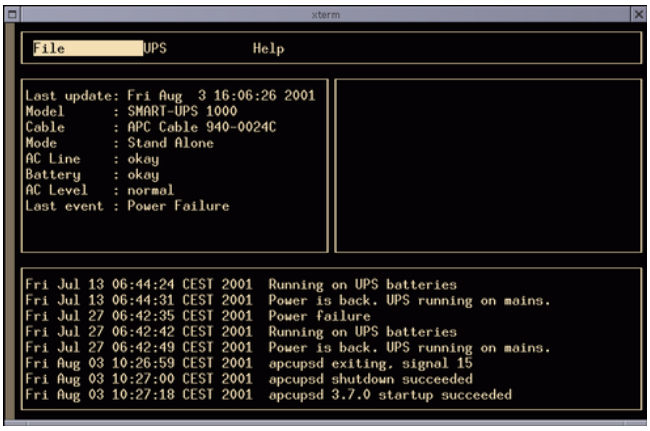
Součástí distribuce **apcupsd** je několik utilit pro monitorování stavu UPS:

- apcaccess: je utilita pro příkazovou řádku, která podrobně vypíše stav a konfiguraci UPS, případně hodnoty parametrů uložených v paměti EEPROM UPS;
- powerflute: je jednoduchá terminálová aplikace pro monitorování stavu UPS;
- webové rozhraní multimon.cgi: umožňuje monitorovat přes síť jednu či více UPS.

Bezpečnost

Pokud se rozhodneme zpřístupnit informace o stavu UPS přes síť, (což je nezbytné např. pro funkčnost CGI rozhraní), tedy se zapnutou direktivou NETSERVER, je třeba počítat s tím, že ve výchozím nastavení může stav UPS číst kdokoliv. Apcupsd od verze 3.8.2 obsahuje podporu balíčku **tcp_wrappers**, takže můžeme přístup omezit patřičnou konfigurací v souborech /etc/hosts.deny a /etc/hosts.allow. Další možností (dostupnou i ve starších verzích apcuspd) je použít místo apcupsd pro publikaci stavu UPS samostatný démon apcnetd, který můžeme startovat přes inetd. (Tedy v konfiguračním souboru apcupsd.conf uvedeme NETSERVER off a nakonfigurujeme inetd tak. aby po příchozím požadavku spojení na port 7000 spouštěl apcnetd. Opět můžeme využít **tcp_wrappers** pro omezení přístupu.)

Pokud provozujeme více počítačů napájených přes jeden záložní zdroj a tyto počítače apcupsd kontroluje přes síť, je potřeba zajistit, aby nebylo možné simulovat falešný **master** apcupsd server, který by pak mohl kontaktovat systémy, které fungují jako **slave** a iniciovat neoprávněně shutdown. V konfiguraci **slave** počítačů je možné direktivou USERMAGIC nastavit heslo, které apcslv pošle při prvním kontaktu **master** procesu apcupsd. Tímto heslem pak **master** proces prokazuje svou totožnost. Ani toto řešení ovšem není ideální, protože heslo po síti putuje v nezašifrované podobě.



Powerflute — nástroj pro konzoli

Shrnutí

Záložní zdroje APC patří mezi ty nejrozšířenější a cenově přístupné. Pod Linuxem jsou dobře podporovány — uživatel

má na výběr mezi software od APC (**PowerChute**) a volně šiřitelným sofware (zejména **Apcupsd** a **NUT**). Může si tedy podle vlastních potřeb vybrat takový software, který mu svým ovládáním nejvíce vyhovuje. Obslužný software **Apcupsd** je co do možností konfigurace velmi flexibilní a pro monitorování stavu UPS poskytuje nástroje jak pro příkazovou řádku / konzoli tak i webové rozhraní. Výhodou je i rozsáhlá a velmi přehledná dokumentace ve formě manuálu (7), článku v Linux Journalu (8), a dalších informací (9). ■

1 American Power Conversion Corporation
http://www.apcc.com/
2 Projekt Apcupsd
http://www.apcupsd.org/
3 FTP server společnosti APC
ftp://ftp.apcc.com/
4 NUT
http://www.exploits.org/nut/
5 genpowerd
ftp://metalab.unc.edu/pub/Linux/system/ups/genpower-1.0.2.tar.gz
6 upsd
ftp://metalab.unc.edu/pub/Linux/system/ups/upsd-2.5.tgz
7 Manuál k apcuspd
http://www.apcupsd.org/users_manual/index.html
8 Článek z Linux Journal o Apcupsd
http://www.apcupsd.org/apcupsd_article_LJ_dec20 00/index.html
9 Další informace o Apcupsd
http://www.sibbald.com/apcupsd/

Zasmáli jsme se!

David Häring

Na závěr jsme připravili opět něco lehčího pro pobavení nejen ze střípků konference *cz.comp.linux* a vašich příspěvků :-)

S problémem synchronizace času se dříve či později setká prakticky každý. Všimněme si, co všechno je třeba vzít v potaz:

> … poslední dobou pořád narážím na nějaké záhady :), tentokrát je to zajímavě synchronizovaný čas pomocí NTP. Používám ntpd dodávaný v RH7.1, mám 2 softwarově a až na videokarty i hardwarově stejné stroje na jednom síťovém segmentu, synchronizované proti stejným serverům. Po celou dobu jejich běhu mají vůči sobě čas posunutý o 40,něco sekund. Ten rozdíl je celou dobu pořád stejný, s přesností řádově milisekund, takže nějaká synchronizace tam je, ale jak to že má každý jiný čas?
> … pretože jeden z nich je určite západnejšie ako ten druhý, a tak má o 40 sekund menej :)
> … No jo, asi o 8 metrů. ;)

O tom, že stařičký netscape mívá s interpretací HTML občas výrazné problémy není pochyb. Někdy je netscape zřejmě natolik bezradný, že se sebekriticky dožaduje instalace patřičného pluginu:

Máte disky Western Digital? Zejména některé série disků této značky byly problematické, ovšem nyní se zdá, že problém má velmi snadné řešení:

> … Stalo se mi, že jsem pustil počítač, a místo naběhnutí se ozývalo hlasité klepání z HD … Znovu jsem provedl instalaci s kontrolou disku a až v průběhu to klepání přestalo. Reklamovat ho už nemůžu, už ho mám asi 3 roky a běží dál jako by nic. Výrobce je myslím Western Digital. Nemyslím si, že je chyba v něm.
> … Mám tu jeden starý 2GB (ST32520A) a taky klepal. Jednou jsem ho vyndal z počítače a chtěl levně prodat, ale spadl mi asi z metrove výšky na zem. Tak jsem ho jen ze zvědavosti dal zpátky do počítače a on už neklepal. Ten disk slouží do dnes :-) Takže rada zní: Nevyhazovat! Omlátit o zeď a zpátky do PC. :-)

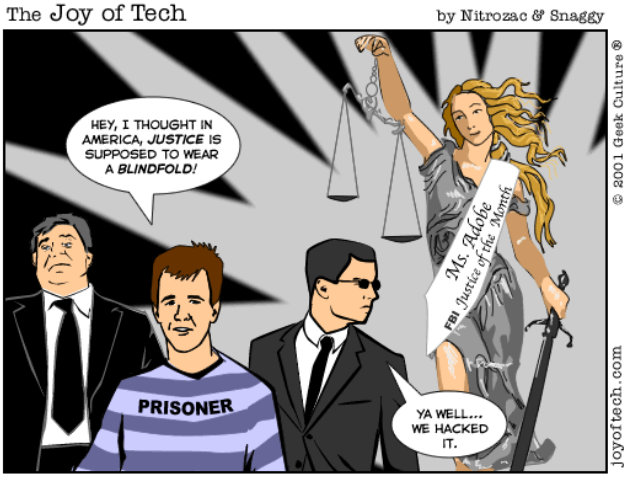
Následuje jednoduchý návod jak zajistit v případě potřeby vzdáleně reset serveru:

> …Dávnejšie tu prebehla debata o zariadení, ktoré umožňuje kontrolovat PC a v prípade že sa sekne ho resetnúť. Môžete mi niekto poslat adresu prípadne nejaké info o tom malom „zázraku“ ? :))
> …kedysi bol vo fide popis od nejakého rusa, ktorému sa nechcelo chodiť cez celý Petrohrad resetovat BBS keď sa mu kúsla. Tak dal oproti sebe dve mašiny s predlženým čudlíkom na reset vždy smerovanom do úrovne cd-mechaniky naprotivného stroja a keď sa kúsol jeden, tak sa konektol na druhý stroj a vyvolal eject príslušnej cd-mechaniky ktorá po vysunuti trafila reset kúsnutého stroja.

Diskuze o tom, jak to vypadá, když se po čase administrátor rozhodne udělat pořádek na serveru a začne přemýšlet, proč je něco nastaveno tak a ne jinak, když by to přece šlo zařídit mnohem lépe:

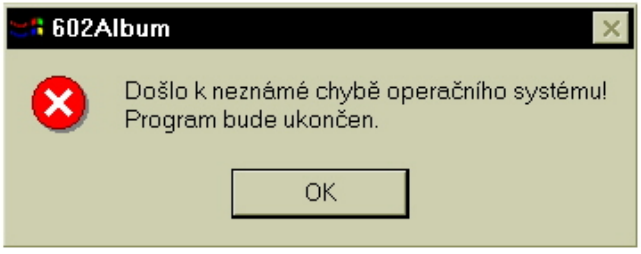
… I pokročilý ale občas může mít problém si vzpomenout, proč je to nastavené *zrovna takhle*.
… Zase se tím cvičí dlouhodobá paměť ;) Člověk na to čučí, říká si ”sakra co to je za nesmysl”, uvede to do (podle něj) správnějšího stavu, provede blablabla restart, zjistí, že přestalo fungovat 482 věcí, VZPOMENE si, proč to teda tak bylo, a zase to vrátí zpátky ;) Nebo ne? :)
…Přesně. Podruhé to je stejné, potřetí už ne, protože se to napíše na papír, který se dobře ukryje (nejlépe do koše). Pak si to člověk napíše do $HOME, který v kritické chvíli smaže.
A pak se řekne: nesahej na to, co funguje — a je (dočasně) po problému :)

Trošku z jiného soudku. Tento kreslený vtip ve skutečnosti vlastně zase tak moc legrační není — je spíše k pláči. O americkém DMCA (Digital Millenium Copyright Act) a skandálech s ním spojených asi lidé pohybující se ve světě Linuxu vědí. Ovšem málo se ví, že spravedlnost už v USA dávno není slepá, ale své oběti si pečlivě vybírá:



Dmitry Sklyarov in the Land of the Free, and the Home of the Dumb Millennium Copyright Act.

A teď ke konkurenci. Linuxu bývá začátečníky vyčítáno, že jádro sice produkuje spoustu podrobných chybových hlášení, ale je problém se v nich vyznat. To takové MS Windows(TM) uživatele rozhodně žádnými zbytečně podrobnými údaji nezatěžují, některé chyby je prostě dokonale překvapí:



Historie UN*Xu je poměrně nepřehledná, a tak se občas stane, že až po uplynutí dlouhé doby na povrch vyplavou zajímavé souvislosti. Na jednu takovou nás upozornil Daniel Novotný:

Hyperuživatel odhalen

Bylo odhaleno šokující tajemství týkající se UNIXových systémů: všichni jistě víte, že superuživatel se jmenuje root podle slečny Rút, milenky Briana Kerninghana. Průhledná záminka, že je to anglicky ”kořen”, určitě neobstojí. Co však bylo odhaleno až nyní, je existence hyperuživatele.

Jméno hyperuživatele je pokračováním v tradici dívčích hebrejských jmen: nazývá se „rivka“. Jméno je napevno zakodováno do kernelu a programu /bin/login. Samozřejmě v zašifrované podobě, jak jinak. Heslo se našim reportérům bohužel nepodařilo zjistit, ale pokud se někdo přihlásí pod hyperuživatelskou identitou, získá tyto výhody:

- procesor se přepne do speciálního nedokumentovaného režimu — na PC tomu odpovídá zvláštní obdoba System Management Mode: program vidí celou paměť, může ke všem portům a může používat nedokumentované instrukce jako HCF (Halt and Catch Fire)
- program cat(1) má nyní novou volbu -undelete, pomocí které je možno bezpečně obnovovat smazané soubory
- díky aktivaci rezervovaných oblastí disku se zvýší úložný prostor
- dd(1) má nyní novou volbu: dd -game spustí hru DoomDestroyer, což je předchůdce Doomu ze sedmdesátých let, běžící v textovém režimu
- přičemž hyperuživatel zároveň samozřejmě může vše co superuživatel.

Díky konspirační domluvě mezi Kerninghanem, Stallmanem a Torvaldsem je hyperuživatel skryt i v Linuxu: někde v hloubi zdrojáků jádra a /bin/login, mezi vším tím assemblerem a céčkem. Můžete ho (vlastně ji) najít čtením zdrojových textů. Přeji vám hodně štěstí. Ovšem to, že vznikne server www.rivka.cz je skutečně pouhá kachna, nevěřte tomu…

Klasický „flame“, aneb co je lepší: Vim nebo Emacs? No, jak pro koho :)

Linuxové noviny a jejich šíření

Pavel Janík

Linuxové noviny vydává České sdružení uživatelů operačního systému Linux (1) pro své příznivce a sympatizanty. Vlastníkem autorských práv k tomuto textu jako celku je Pavel Janík ml. (Pavel.Janik@linux.cz). Autorská práva k jednotlivým článkům zůstávají jejich autorům.

Tento text může být šířen a tištěn bez omezení. Pokud použijete část některého článku zde uveřejněného v jiných dílech, musíte uvést jméno autora a číslo, ve kterém byl článek uveřejněn.

Linuxové noviny jsou otevřeny každému, kdo by chtěl našim čtenářům sdělit něco zajímavého. Příspěvky (ve formátu čistého textu v kódování ISO 8859-2) posílejte na adresu (2). Autor nemá nárok na finanční odměnu a souhlasí s podmínkami uvedenými v tomto odstavci. Vydavatelé si vyhrazují právo rozhodnout, zda Váš příspěvek uveřejní, či nikoli.

Registrované známky použité v tomto textu jsou majetkem jejich vlastníků.

Chtěl bych poděkovat společnosti SuSE CR, s.r.o (3), že podporuje redakci v práci na Linuxových novinách.

Linuxové noviny jsou k dispozici také ve formátu HTML na adrese (4). ■

1 České sdružení uživatelů operačního systému Linux
http://www.linux.cz/czlug
2 Adresa redakce
mailto:noviny@linux.cz
3 SuSE CR, s.r.o.
http://www.suse.cz/
4 Linuxové noviny ve formátu HTML
http://www.linux.cz/noviny

Šéfredaktor: Pavel Janík ml.
mailto:Pavel.Janik@linux.cz
výkonný šéfredaktor: David Häring
mailto:dave@ibp.cz
sazba: Ondřej Koala Vácha
mailto:koala@informatics.muni.cz