ČÁST I

Úvod do Linuxu
Co je to Linux?

Začneme historickým přehledem toho, jak se Linux stal tím operačním systémem, jímž je dnes. Budeme hovořit o minulosti a budoucnosti jeho vývoje a blíže se podíváme na výhody a nevý-hody tohoto systému. Zmíníme se o distribucích, obecně o filozofii otevřeného zdrojového kódu a pokusíme se vysvětlit základní myšlenky licence GNU.

V této kapitole odpovídáme na otázky, jako jsou:

Co je to Linux?
Kde a jak Linux začal?
Není Linux ten systém, kde se všechno dělá v textovém režimu?
Má Linux budoucnost, nebo je to jen bublina?
Jaké jsou výhody používání Linuxu?
A jaké jsou nevýhody?
Jaké druhy Linuxu existují a jak vybrat ten, co mi nejvíce vyhovuje?
Co je to hnutí Open Source a GNU?

Historie

UNIX

Chceme-li pochopit popularitu Linuxu, musíme se přesunout zpátky v čase, do doby asi před 30 lety... Představte si počítače velké jako domy, jako celé budovy. I když problém představovala sama veli-kost počítačů, byl zde ještě další, mnohem větší problém: Každý počítač používal vlastní operační systém. Software se vždy vytvářel na míru konkrétním potřebám a software pro jeden počítačový systém nefungoval na jiném systému. Když jste uměli pracovat na jednom systému, neznamenalo to, že můžete pracovat i na jiném. Bylo to těžké, jak pro uživatele, tak pro administrátory.

Počítače byly mimořádně drahé a i po jejich zakoupení jste museli dále platit například za to, aby se uživatelé mohli dozvědět, jak počítač funguje. Cena za jednotku výpočetního výkonu byla astronomická.

Technologie té doby nebyla ještě dostatečně vyspělá, takže bylo nutné se s velikostí počítačů smí-řit ještě na celých deset let. Mezitím, v roce 1969, začal tým výzkumníků v Bellových laboratořích pracovat na řešení softwarového problému. Vyvinuli nový operační systém, který byl:

1Jednoduchý a elegantní.
2Napsán v jazyce C, nikoliv v assembleru.
3Schopen recyklace kódu.

Vývojáři v Bellových laboratořích tento projekt pojmenovali UNIX. Schopnost recyklace kódu byla nesmírně důležitá. Do té doby se všechny komerčně dostupné počítačové systémy programovaly v jazyce vytvořeném speciálně pro daný systém. Naproti tomu UNIX potřeboval jen malou část tohoto specializovaného kódu, část, které dnes běžně říkáme jádro. Jádro je jediná část kódu, která musí být vytvořena speciálně pro hardware konkrétního systému, a představuje základ operačního systému UNIX. Operační systém a všechny ostatní funk-ce jsou postaveny okolo jádra a jsou naprogramovány ve vyšším programovacím jazyce, v jazyce C. Tento jazyk byl vytvořen právě pro potřeby vývoje systému UNIX. Díky této nové technice bylo mnohem snazší vyvinout operační systém, který bude schopen pracovat na různých typech hardwaru.

Výrobci softwaru se přizpůsobili velmi rychle, protože byli najednou schopni s minimálním úsilím prodat desetinásobné množství programů. Začaly se objevovat nové neobvyklé situace: Například počítače různých výrobců komunikující ve stejné síti anebo

uživatelé, kteří mohli pracovat na různých systémech, aniž by pro každý potřebovali specializované školení. UNIX tak uživatelům nabídl kompatibilitu mnoha různých systémů.

Dalších dvacet let vývoj systému UNIX pokračoval. Dalo se dělat více a více věcí a stále více výrob
ců hardwaru i softwaru ve svých produktech podporovalo UNIX. Zároveň se ale začaly objevovat i menší počítače a koncem 80. let měla spousta lidí domácí počí-tač. V té době existovalo pro platformu PC několik variant UNIXu, ale žádná z nich nebyla snad-no dostupná a navíc byly všechny hrozně pomalé, takže většina lidí na svých domácích počíta-čích používala MS-DOS nebo Windows 3.1.

Linus a Linux

Na počátku 90. let byl výkon domácích počítačů konečně postačující k tomu, aby na nich mohl běžet plnohodnotný UNIX. V té době Linus Torvalds, student počítačových věd na univerzitě v Helsinkách, usoudil, že by bylo vynikající, kdyby existoval nějaký typ volně dostupné akade-mické verze systému UNIXu a hned na tom začal pracovat.

Začal se ptát, hledal odpovědi a možnosti, jak dostat UNIX na své PC. Takto vypadal jeden z jeho prvních příspěvků v diskusní skupině comp.os.minix z roku 1991 (hlavička zkrácena):

From: torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)Newsgroups: comp.os.minixSubject: Gcc-1.40 and a posix-questionMessage-ID: <1991Jul3.100050.9886@klaava.Helsinki.FI>Date: 3 Jul 91 10:00:50 GMT

Ahoj,
kvůli projektu, na němž pracuji (v Minixu), by mě zajímala definice standardu
POSIX. Můžete mě někdo navést na (nejlépe elektronickou verzi) nejnovější spe
cifikace Posixu? Výborný by byl např. FTP server.

Od počátku bylo Linusovým cílem vytvořit volně dostupný systém plně kompatibilní s původním systémem UNIX. Právě proto sháněl definici standardu POSIX, což je do dnešní doby používaná standardní specifikace UNIXových systémů.

Technologie plug-and-play v té době ještě neexistovala, ale vlastní UNIX chtěla mít na svém počí-tači spousta lidí, takže to nebyla žádná překážka. Stále rychleji se objevovaly nové ovladače pro nové hardwarové komponenty. Jakmile se objevilo nějaké nové zařízení, téměř okamžitě si je někdo koupil a začal je testovat v Linuxu, jak se novému systému začalo říkat. Volně dostupné ovladače tak vznikaly pro stále širší množinu hardwaru. Tito programátoři se navíc nezastavili jen
u svého počítače – snažili se v Linuxu oživit jakýkoliv kus hardwaru, na který narazili. Seznam podporovaných hardwarových komponent se tak neustále rozrůstal. Díky těmto nadšen-cům je tak dnešní Linux vhodný nejen na nejnovější počítače, ale můžete jej použít i se starými a exotickými hardwarovými komponentami, které by bez Linuxu byly dnes k ničemu.

Za dva roky od Linusova oznámení měl Linux 12 000 uživatelů. Projekt oblíbený mezi počítačovými nadšenci se trvale rozrůstal, přičemž se stále držel standardu POSIX. V průběhu několika let byly doplněny všechny funkce unixových systémů, takže výsledkem je dnešní vyzrálý operační systém Linux. Linux je plnohodnotný klon systému UNIX, použitelný jak na pracovních stanicích, tak na středních a špičkových serverech. Většina významných hráčů na dnešním hardwarovém a softwarovém trhu má tým linuxových vývojářů a dnes už není problém koupit si v počítačové prodejně sestavu s nainstalovaným Linuxem a zajištěnou podporou. Samozřejmě stále existuje velká skupina nepodporovaného softwaru i hardwaru.

Současné použití linuxových systémů

V současnosti je Linux součástí trhu se stolními počítači. Zpočátku se vývojáři Linuxu zaměřovali na síťové systémy a služby, kancelářské aplikace představovaly poslední bariéru, kterou bylo nutno překonat. Neradi přiznáváme, že tomuto trhu dominuje Microsoft, proto v posledních něko-lika letech vznikla spousta alternativních projektů, jejichž cílem je nabídnout Linux jako vhodnou volbu na pracovní stanice. V rámci těchto projektů vznikají snadno použitelná uživatelská rozhra-ní i kancelářské aplikace, jako textové editory, tabulkové procesory a podobně, kompatibilní s aplikacemi Microsoft Office.

Co se týče použití na serverech, má Linux pověst stabilní a spolehlivé platformy, na které běží databázové a další služby takových společností, jako je Amazon, známé elektronické knihkupec-tví, americká pošta, německá armáda a další. Velmi oblíbený je Linux u poskytovatelů interneto-vého přístupu a služeb, kde se používá jako firewall, proxy server nebo webový server. Počítač s Linuxem najdete i u každého správce některého UNIXového systému, který jej používá jako pohodlnou administrativní stanici. Clustery linuxových počítačů se podílely na vzniku filmů jako Titanic nebo Shrek. Na poštách slouží jako centrály řídící směrování zásilek, velké vyhledávací stroje pomocí nich prohledávají Internet. To je jen ukázka několika z mnoha tisíc náročných úkolů, které dnes Linux na celém světě vykonává.

Stojí také za zmínku, že moderní Linux běží nejen na pracovních stanicích, středních a velkých serverech, ale i na „hračkách“, jako jsou PDA či mobilní telefony, ve spoustě zařízení spotřební elektroniky, a dokonce i v experimentálních náramkových hodinkách. Linux je jediným operačním systémem na světě, který pokrývá takto širokou škálu hardwaru.

Uživatelské rozhraní

Je Linux těžký?

Odpověď na otázku, zda je těžké se Linux naučit, bude záviset na tom, koho se zeptáte. Zkušení uživatelé systému UNIX vám odpovědí, že ne, protože Linux je ideální operační systém pro zku-šené uživatele a programátory, jelikož byl a je takovýmito lidmi vyvíjen.

Cokoliv si může zkušený programátor přát, to v Linuxu najde: překladače, knihovny, vývojové a ladi-cí nástroje. Tyto komponenty jsou součástí každé standardní linuxové distribuce. Překladač jazyka C je k dispozici zdarma – na rozdíl od mnoha UNIXů, kde je za tento nástroj nutno platit zvlášť. K dis-pozici je veškerá dokumentace a manuály, které často obsahují i příklady, díky nimž můžete oka
mžitě začít pracovat. Vypadá to tak, že přechod od systému UNIX k Linuxu je přirozená věc. V dřívějších linuxových dobách musel být člověk expert už jenom proto, aby systém nainstaloval a mohl začít používat. Tehdejší uživatelé Linuxu se cítili nadřazeni ostatním uživatelům, kteří dosud „nespatřili světlo“. Běžná odpověď na dotazy začínajících uživatelů zněla „RTFM“ (přečti si manuál). Manuál sice byl součástí každého systému, nalezení potřebné dokumentace ale nebylo snadné, a i když to člověk zvládl, dokumentace byla psána natolik odborně, že to začínajícího uživatele velmi rychle odradilo.

Komunita uživatelů Linuxu ovšem postupně zjistila, že pokud se má pozice Linuxu na trhu operačních systémů posílit, je nutné použitelnost systému zásadně změnit.

Linux pro začínající uživatele

Problém začaly řešit společnosti jako RedHat, SuSE (dnes součást Novellu) a Mandriva (dříve Man-drakeSoft), které začaly vytvářet předpřipravené distribuce Linuxu, vhodné k masové spotřebě. Součástí distribuce byla řada grafických uživatelských nástrojů, které usnadňovaly správu progra-mů a služeb. Současný uživatel Linuxu má stále možnost seznámit se se svým systémem do nejmenších podrobností, na druhé straně ale takové znalosti nepotřebuje k běžnému užívání systé-mu.

Dnes se rovnou přihlásíte do grafického prostředí a všechny potřebné aplikace spustíte pomocí myši, stále však v případě potřeby můžete manipulovat s interními detaily systému. Díky tomuto členění má uživatel možnost postupného růstu a systém vyhovuje jak začínajícím, tak i zkušeným uživatelům. Noví uživatelé nejsou nuceni začínat hned s obtížnými věcmi, zkušení uživatelé nemu-sí trvale používat systém stejným způsobem, jako když s ním začínali.

Vývoj v oblasti systémových služeb stále pokračuje, zároveň se ale dělá hodně pro řadové uživa-tele, u nichž se obecně předpokládá, že je nijak nezajímá, jak systém funguje uvnitř. Vývojáři gra-fických aplikací věnují mimořádné úsilí vytváření nádherně vyhlížejících systémů, stejně tak ale nabízejí možnost, aby linuxový systém vypadal opticky úplně stejně jako MS Windows nebo Macintosh. V poslední době se usiluje i o podporu 3D grafické akcelerace, USB zařízení, nástrojů pro instalaci a aktualizaci programů „jedním kliknutím myši“ a podobně. To vše dnes Linux nabí-zí a snaží se všechny dostupné služby představit v logické podobě, která bude pochopitelná pro běžného uživatele. Následující krátký seznam nabízí několik skvělých příkladů. Na příslušných stránkách najdete mnoho ukázek, které vám dají představu o tom, jak může linuxová pracovní sta-nice vypadat:

http://www.gnome.org
http://kde.org/screenshots/
http://www.openoffice.org
http://www.mozilla.org

Má Linux budoucnost?

Hnutí Open Source

Základní myšlenka hnutí Open Source, tedy programů s otevřeným kódem, je velmi jednoduchá: Má-li programátor možnost číst, měnit a distribuovat kód programu, má z toho program užitek. Uživatelé mohou program přizpůsobovat, opravovat a testovat, a to vše rychlostí, která je nesrov-natelná s tempem vývoje v klasických softwarových společnostech. Takto vzniklé programy budou univerzálnější a lepší než tradičně vzniklé programy, protože je otestovalo mnohem více lidí
a v mnohem různorodějších podmínkách, než je v silách jakékoliv softwarové společnosti. Iniciativa Open Source se tuto myšlenku snaží vštípit komerčnímu světu a komerční výrobci soft-waru ji začínají, ovšem velmi pomalu, chápat. Zatímco lidem z akademické a technické sféry je již 20 let jasné, že toto je ta správná cesta, komerční výrobci potřebovali takové aplikace, jako je Inter-net, aby jim ukázal, že i oni mohou z Open Source těžit. V současné době již Linux překročil hra-nici převážně akademického systému, vhodného pouze pro hrstku lidí s technickým zázemím. Linux je dnes mnohem více než jen operační systém: Existují infrastruktury, které podporují jeho vývoj, vývoj a testování programů pro Linux, ale také zajišťují úplný servis koncovým uživatelům

– tedy údržbu, aktualizaci, podporu a specifické přizpůsobení. Linux je dnes plně schopen čelit výzvám rychle se měnícího světa.

Deset let zkušeností k vašim službám

Linux je zřejmě nejznámějším výsledkem iniciativy Open Source, na jeho současné popularitě má ovšem zásluhu i jiný projekt. Tento projekt se jmenuje SAMBA a jeho výsledkem je úplná reverz-ní analýza protokolů SMB (Server Message Block) a CIFS (Common Internet File System), což jsou protokoly používané pro sdílení souborů a tiskáren mezi osobními počítači, podporované nativ-ně v systémech MS Windows a OS/2. Dnes už jsou tyto protokoly dostupné pro prakticky jaký-koliv systém a nabízejí možnost propojit systémy v heterogenním prostředí pomocí protokolů MS Windows, takže je možno sdílet soubory a tiskárny se všemi verzemi systémů Windows.

Možná ještě úspěšnějším projektem než SAMBA je projekt HTTP serveru Apache. Tento server běží na systému UNIX, Windows NT a mnoha dalších operačních systémech. Původní interpretace názvu zněla „A PAtCHy server“, což vycházelo z nutnosti „patchovat“ (čili aplikovat opravy na) základní kód spoustou doplňků. Název má ovšem ve skutečnosti narážet na původní indiánský kmen Apačů, vyhlášený svou vynikající bojovou strategií a nevyčerpatelnou vytrvalostí. Dnes je Apache podstatně rychlejší, stabilnější a nabízí mnohem více funkcí než celá řada jiných serverů. Apache běží v systémech s miliony návštěvníků denně, a i když vývojáři oficiálně neposkytují žád-nou podporu, uživatelská komunita tohoto serveru je schopna odpovědět na jakoukoliv otázku. Celá řada firem navíc nabízí komerční podporu.

Na poli kancelářských aplikací vznikla řada klonů balíku MS Office, od částečných implementací až po úplné implementace všech aplikací dostupných v MS Windows. Tyto iniciativy se význam-ně zasloužily o nástup Linuxu na pracovních stanicích, protože uživatelé nepotřebují žádné zvlášt-ní školení k tomu, aby mohli s novým systémem pracovat. S nasazením na desktopech přichází i noví spokojení řadoví uživatelé a s nimi jejich specifické požadavky, které jsou dnes stále složi-tější a náročnější.

Komunita Open Source, jejíž většina členů přispívá více než pět let, zajišťuje Linuxu pozici významného hráče jak na poli pracovních stanic, tak na obecném trhu IT aplikací. Udržení této pozice společně zajišťují jednak placení vývojáři, jednak dobrovolníci. S nárůstem počtu uživatelů roste i počet otázek. Komunita nabízí odpovědi a zároveň pečlivě sleduje, jaké tyto odpovědi jsou, což se dále promítá do zvýšení stability a použitelnosti.

Výčet všech programů, které jsou pro Linux k dispozici, je mimo záběr této příručky, protože soft-warových balíků existují desítky tisíc. V tomto textu se budeme zabývat těmi nejznámějšími, z nichž většina je k dispozici zdarma. Abychom zahnali obavy začínajících uživatelů, uvádíme obrázek jednoho z nejžádanějších programů. Jak můžete vidět, nebylo vynecháno nic, aby se uži-vatelé přicházející z Windows mohli cítit jako doma:

Vlastnosti Linuxu

Pro

Řada výhod Linuxu vyplývá z jeho původu, založeného na UNIXových systémech – samozřejměs výjimkou první výhody:
Linux je zdarma:
Pokud nebudete chtít utratit vůbec nic, nemusíte platit ani za CD*. Celý Linux můžete zdarma stá-hnout z Internetu. Žádné registrační poplatky, žádné licenční poplatky za každého uživatele, aktu-alizace zdarma, a pokud byste chtěli systém modifikovat, zdarma dostupné zdrojové kódy.
Z velké části je Linux také svobodný:Nejčastěji používanou licencí je GNU Public Licence (GPL). Tato licence říká, že kdokoliv budechtít, má právo Linux modifikovat a změněnou verzi může dále distribuovat s jedinou podmínkou

* Poznámka českého vydavatele: Uvedená věta samozřejmě platí, ale Linux nemusí být vždy zdarma. Například některé distribuce Linuxu, viz dále, se kupují. Mnoho lidí považuje za dobrý nápad živit se Linuxem – proto jej třeba prodávají – a stejně tak mnoho lidí tohle považuje za dobrý nápad, a proto si Linux kupují. Chcete-li vědět více o ceně Linuxu a free softwaru, přečtěte si dokument na adrese *http://proc.linux.cz/kde-ziskat.html*, případně trochu srozumitelnější vysvětlení na *http://www.linuxexpres.cz/jak-linux-ziskat.*

– že dá k dispozici změněný kód. V praxi to znamená, že si můžete stáhnout zdrojový kód jádra, doplnit do něj nástroje třeba pro teleportaci nebo cestování časem, upravený kód můžete prodá-vat, ovšem zákazníkům musíte dát k dispozici zdrojový kód provedených změn.
Linux je přenositelný na jakoukoliv hardwarovou platformu:Výrobce, který chce prodávat nový typ počítače a může se rozhodnout, jaký operační systém naněm poběží (může jít například o počítač v autě nebo v myčce na nádobí), může vzít linuxové jádro a upravit je tak, aby běželo na novém hardwaru. Dokumentace potřebná k této úpravě jedostupná zdarma.Linux dokáže běžet trvale:Stejně jako u systému UNIX, i linuxový systém počítá s tím, že poběží dlouhodobě bez restarto-vání. Řada úloh se tak může provádět v noci nebo se automaticky plánují na dobu s nižším pro-vozem, což vede k lepší dostupnosti systému v době vysokého zatížení a rovněž k rovnoměrněj-šímu využití hardwaru. Tato vlastnost vede rovněž k tomu, že Linux lze použít v prostředích, kdečlověk nemá čas nebo možnost běh systému trvale sledovat.
Linux je bezpečný a univerzální: Bezpečnostní model Linuxu vychází z unixových bezpečnostních principů, které jsou považová-ny za robustní a prověřené. Linux se dá ovšem použít nejen jako ochrana před útoky z Internetu

– stejně dobře poslouží i v jiných situacích a nabídne stejně vysoký bezpečnostní standard. Vývojový počítač nebo řídící stanice budou zabezpečeny stejně jako firewall.
Linux je škálovatelný: Můžete mít palmtop se 2 MB paměti anebo petabajtový diskový cluster se stovkami uzlů: Přidejte nebo odeberte potřebné balíčky, a Linux poběží na obou. Nepotřebujete superpočítač, náročné úkoly může plnit i Linux s použitím stavebních kamenů, které standardně obsahuje. A pokud chcete něco malého, například embedded systém nebo jen reinkarnovat starou 486, Linux to zvládne rovněž.
Operační systém a některé aplikace mají velmi rychlou reakci na chyby:Protože Linux vyvíjejí a testují tisíce lidí, obvykle se velmi rychle najde chyba i někdo, kdo jiodstraní. Není neobvyklé, že od ohlášení chyby do jejího odstranění uplyne jen několik hodin.

Proti

Existuje příliš mnoho různých distribucí:
Staří Římané říkali „quot capites, tot rationes“, více lidí, více názorů. Na první pohled vás může počet linuxových distribucí polekat anebo udivit – záleží na úhlu pohledu. Na druhé straně to ale

znamená, že každý může najít to, co mu bude vyhovovat. K nalezení vhodné distribuce nepotře bujete být žádný expert.
Zeptáte-li se běžného linuxového uživatele na nejlepší distribuci, většina vám doporučí právě tu,

kterou sami používají. Kterou distribuci tedy vybrat? Nemusíte se tím příliš zabývat: Většina distri-bucí obsahuje víceméně stejnou základní množinu komponent. K těmto základům přibývají spe-cializované programy třetích stran, díky nimž je například TurboLinux vhodnější na malé a střed-ní aplikace, RedHat na servery a SuSE na pracovní stanice. Tyto rozdíly jsou ovšem vesměs jenokrajové. Nejlepší metoda je vyzkoušet více distribucí – bohužel ne každý má na to čas. Naštěstík problematice výběru vhodné distribuce existuje spousta doporučení. Zkuste se například zeptatGooglu na „choosing Linux distribution“ – objeví se desítky odkazů vedoucích na užitečné rady.O volbě distribuce hovoří rovněž dokument Installation HOWTO.
Linux není dostatečně „uživatelsky přítulný“ a pro začátečníka je matoucí:Je nutno říct, že Linux, alespoň na úrovni základního jádra, je z pohledu uživatele méně přátel-ský než MS Windows a je určitě obtížnější než MacOS, ale... Vzhledem k rostoucí popularitě sezvyšuje snaha co nejvíce usnadnit používání Linuxu zejména novým uživatelům. Neustále vznika-
jí nové návody, včetně této příručky, které se snaží zaplnit mezery v dokumentaci pro uživatele
na všech úrovních.
Lze Open Source produktu důvěřovat?
Může být něco, co je zadarmo, zároveň spolehlivé? Uživatelé Linuxu měli možnost rozhodnout se,

zda Linux budou, či nebudou používat, což je velká výhoda v porovnání s uživateli proprietárníchprogramů, kteří tuto volbu neměli. Po delší době testování došla většina uživatelů Linuxu k závě-ru, že Linux je nejen dobrý, ale v řadě případů i lepší než tradiční řešení. Pokud by Linux nebyldůvěryhodný, zmizel by už před dlouhou dobou a nikdy by nedosáhl své současné popularitya milionů uživatelů. Uživatelé mohou svůj systém ovlivnit a sdílet své postřehy s komunitou, takžese systém denně vylepšuje. Je pravda, že jde o nikdy nekončící projekt, ovšem v prostředí, kterése trvale mění, je Linux projektem, který stále usiluje o dokonalost.

Varianty Linuxu

Linux a GNU

Přestože existuje mnoho různých implementací a distribucí Linuxu, najdete mezi nimi celou řadu podobností, přičemž jedna z nich vždy bude spočívat v tom, že linuxový systém je vždy sestaven z komponent, které si můžete vybírat podle svých potřeb a požadavků. Instalace systému je pouhý počátek dlouhodobého vztahu. Vždy když budete mít příjemný pocit z dobře běžícího systému, Linux dokáže stimulovat vaši představivost a kreativitu, a čím více budete znát možnosti svého systému, tím více budete chtít tyto limity překročit.

V závislosti na použité distribuci, hardwaru a osobním vkusu může Linux vypadat pokaždé jinak, nicméně základ, na němž jsou grafická a všechna další rozhraní vybudována, zůstává pořád stej-ný. Linuxový systém je založen na nástrojích GNU, které nabízejí standardní způsoby pro práci a manipulaci se systémem. Všechny nástroje GNU mají otevřený zdrojový kód, takže mohou být nainstalovány na jakémkoliv systému. Většina distribucí nabízí připravené balíčky s obvyklými nástroji, ať už jsou to balíčky RPM systému RedHat nebo balíčky deb či dpkg systému Debian – abyste ve svém systému nainstalovali potřebný program, nemusíte být programátor. Pokud ovšem mezi programátory patříte a baví vás zařídit si vše po svém, užijete si Linux také, protože většina distribucí obsahuje všechny potřebné vývojové nástroje a máte tak možnost nainstalovat nový soft-ware čistě

ze zdrojových kódů. Díky tomu můžete nainstalovat i programy, které pro vámi pou-žívanou distribuci neexistují v podobě připraveného balíčku.

Následující seznam uvádí nejznámější programy z rodiny GNU:

Bash: GNU shell
GCC: GNU překladač jazyka C
GDB: GNU debugger
Coreutils: základní nástroje unixového typu, například ls, cat a chmod
Findutils: nástroje pro hledání a prohledávání souborů
Fontutils: nástroje pro konverzi fontů z jednoho formátu do druhého a pro vytváření nových fontů.
The GIMP: program pro manipulaci s rastrovou grafikou
Gnome: grafické uživatelské prostředí
Emacs: velmi mocný textový editor
Ghostscript a Ghostview: interpretr a grafické rozhraní k souborům PostScript
GNU Photo: software pro práci s digitálními fotoaparáty
Octave: programovací jazyk určený primárně pro numerické výpočty a zpracování obrazu
GNU SQL: relační databázový systém
Radius: server pro vzdálenou autentizaci a účtování

Pro Linux existuje i celá řada komerčních programů. Co se týče podrobností o těchto programech, odkazujeme vás na jejich dokumentaci. V této příručce budeme hovořit pouze o zdarma dostupných programech, distribuovaných (ve většině případů) pod licencí GNU GPL.

Abyste mohli instalovat chybějící či nové balíčky, potřebujete nějaký mechanismus pro správu pro-gramů. Nejznámější implementace jsou RPM, dpkg a Ximian Red Carpet. RPM je RedHat Package Manager a používá se v celé řadě linuxových systémů, i když to jeho název nenaznačuje. Dpkg je systém správy balíčků z Debianu, jehož uživatelské rozhraní se jmenuje apt-get a dokáže zpra-covávat i balíčky RPM. Ximian Red Carpet je samostatná implementace systému RPM s grafickým rozhraním. Různé další programy různých dodavatelů mohou obsahovat vlastní instalační mecha-nismus, který může připomínat programy jako InstallShield, známé z MS Windows a dalších plat-forem. Jakmile se s Linuxem blíže seznámíte, velmi pravděpodobně přijdete s některým z těchto systémů do styku.

GNU/Linux

Linuxové jádro (kostra systému, viz kapitolu „Jádro“) není součástí projektu GNU, používá však stejnou licenci jako programy z tohoto projektu. Většina ostatních nástrojů a vývojových kompo-nent (tedy „maso“ systému) není specificky linuxová a pochází z projektu GNU. Jakýkoliv použi-telný systém musí obsahovat jak jádro, tak i alespoň minimální množinu dalších nástrojů – proto se občas uvádí, že celý systém by měl být označován jako GNU/Linux.

Abychom udrželi co největší míru nezávislosti na konkrétních distribucích, budeme v dalším textu hovořit právě o tomto „typu“ Linuxu. Pokud budeme popisovat něco mimo GNU/Linux, uvedeme konkrétní distribuci, verzi či program, o nichž hovoříme.

Jakou distribuci si mám nainstalovat?

Nejdůležitějším faktorem, který předchází instalaci, je používaný hardware. Každá linuxová distri-buce obsahuje základní balíčky a lze ji sestavit pro potřeby téměř jakéhokoliv prostředí (všechny totiž používají stejné linuxové jádro) – potřebujete proto zvolit takovou distribuci, která vašemu hardwaru odpovídá. Například LinuxPPC poběží na Macintoshi a jiných PowerPC systémech, nepoběží na obvyklém PC s platformou x86. LinuxPPC funguje na novějších Macích, nelze jej ale použít na systémech se starou sběrnicovou technologií. Dalším specifickým příkladem je hardwa-re společnosti Sun, který může používat buď starší CPU SPARC nebo novější UltraSparc, přičemž každá varianta vyžaduje jinou verzi Linuxu.

Některé linuxové distribuce jsou optimalizovány pro určité procesory, například Athlon, nicméně budou fungovat i na standardních procesorech Intel 486, 586 a 686. Distribuce určené pro exo-tické procesory mohou být méně stabilní než typické distribuce, protože je testuje menší počet lidí.
Většina linuxových distribucí obsahuje sadu programů určenou pro „obecné PC“ a navíc speciál-ní balíčky optimalizované pro různé procesory rodiny Intel x86. Tyto distribuce bývají dobře otes-tované a pravidelně udržované, protože se zaměřují na spolehlivé nasazení na serverech a snad-nou instalaci a aktualizaci. Příkladem takových distribucí jsou Debian, Ubuntu, Fedora, openSUSE a Mandriva, což jsou jednoznačně nejpopulárnější linuxové distribuce. Obecně jsou považovány za dostatečně jednoduché i pro začínající uživatele, aniž by přitom zkušenému uživateli znemož-ňovaly využít svůj systém na maximum. Linux spolehlivě běží na přenosných počítačích i na ser-verech střední třídy. Ovladače nových hardwarových komponent bývají do systému přidávány až po důkladném otestování, což přispívá k celkové stabilitě systému.

Na některých systémech se jako standardní grafické prostředí používá Gnome, jiné systémy standardně nabízejí KDE. Obecně všechny hlavní distribuce umožňují používat jak Gnome, tak KDE. Většina z nich nabízí i jiné správce oken a správce pracovní

plochy, které jsou určeny pro zkušenější uživatele.

V rámci standardního instalačního procesu si uživatel může zvolit mezi různými základními kon-figuracemi, jako je například „pracovní stanice“, která obsahuje všechny balíčky potřebné pro kaž-dodenní práci a vývoj, nebo „server“, kde se instalují různé síťové služby. Zkušený uživatel si může v průběhu instalace navolit jakoukoliv kombinaci balíčků.

Snahou této příručky je, aby byla nezávislá na konkrétní distribuci. Začínajícím uživatelům však rozhodně doporučujeme, aby si zvolili některou z „hlavních“ distribucí, které podporují veškerý obvyklý hardware a obsahují všechny potřebné aplikace. Vhodnými distribucemi pro začátečníky jsou:

Fedora Core
Debian
■ openSUSE (dříve SuSE Linux)

Mandriva Linux (dříve Mandrake Linux)
Knoppix: operační systém, který běží přímo z CD-ROM, není nutné nic instalovat

Obrazy instalačních CD je možno stáhnout z LinuxISO.org. Běžné distribuce je také obvykle možné koupit v každém slušném počítačovém obchodě. Českým uživatelům jistě pomůže již jednou uve-dená stránka *http://www.linuxexpres.cz/jak-linux-ziskat* nebo *http://proc.linux.cz/kde-ziskat.html.*

Shrnutí

V této kapitole jsme se dozvěděli:

Linux je samostatná implementace systému UNIX
Linux je napsán v programovacím jazyce C
„De gustibus et coloribus non disputandum est“ – Každý si může najít svůj Linux
Linux používá nástroje projektu GNU, volně distribuovanou sadu standardních nástrojů pro práci s operačním systémem

Cvičení

Praktické cvičení pro začátečníky: Nainstalujte si na svůj počítač Linux. Přečtěte si instalační manuál své distribuce a/nebo dokument Installation HOWTO a pusťte se do toho.

> Čtěte dokumentaci!
>
> Většina chyb je způsobena tím, že uživatel nečte informace, které se může v průběhu insta-lace dozvědět. První krok k úspěchu spočívá v pečlivém sledování všech zpráv, které insta-lační proces vypisuje.

Ještě PŘED zahájením instalace potřebujete mít jasno v následujících otázkách:

- Která distribuce bude s mým hardwarem fungovat? Pokud máte pochybnosti o kompatibilitě, ověřte si to v dokumentu *http://www.tldp.org/HOWTO/Hardware-HOWTO/index.html.*

Jakou mám klávesnici (počet kláves, rozložení)? Jakou mám myš (sériovou, paralelní, počet tlačítek)? Kolik mám paměti?
Budu instalovat základní pracovní stanici, server nebo budu potřebovat nějaké speciální balíčky?
Budu instalovat z pevného disku, z CD-ROM nebo přes síť? Je nutné kvůli tomu změnit nastavení BIOSu? Budu k instalaci potřebovat bootovací disk?
Budu Linux používat jako jediný systém, nebo budu mít na počítači nainstalován i jiný systém? Jak mám rozdělit diskové oddíly mezi jednotlivé instalované systémy?
Bude počítač připojen k síti? Jaký je jeho název, IP adresa? Jaká je adresa brány a dalších důležitých síťových systémů, s nimiž bude počítač komunikovat?

> Linux počítá s připojením do sítě
>
> Pokud síť nepoužijete nebo ji nastavíte nesprávně, může se spouštění systému významně zpomalit.

Bude počítač pracovat jako brána, směrovač či firewall? (Pokud nad touto otázkou přemýšlíte, pak je odpověď nejspíš záporná.)
Vytvoření oddílů – napoprvé to ponechejte na instalačním programu, podrobněji budeme

> o diskových oddílech mluvit ve třetí kapitole. Zajímají-li vás podrobnosti, měli byste je nalézt v dokumentaci k vašemu konkrétnímu systému. Pokud instalační program nenabízí automatické rozdělení diskových oddílů, není distribuce nejspíš určena pro začátečníky.

Má počítač startovat v textovém, nebo grafickém režimu?
Zvolte si vhodné heslo správce počítače (root). Kromě toho vytvořte neprivilegovaný účet pro běžnou práci se systémem.

Budu potřebovat záchranný disk? (Doporučujeme.)
Jakou jazykovou mutaci chci instalovat?

Úplný seznam jednotlivých před-instalačních kroků a rozhodnutí naleznete v dokumentu *http://www.tldp.org/HOWTO/Installation-HOWTO/index.html.*

V následujících kapitolách zjistíme, zda se vám instalace podařila.

První kroky

Abychom vám mohli nabídnout co nejužitečnější příručku, začneme rovnou prakticky zaměřenou kapitolou, v níž se budeme věnovat přihlášení k systému a některým základním operacím. Budeme hovořit o těchto tématech:

Přihlášení k systému.
Odhlášení od systému
Textový a grafický režim
Změna hesla
Prohlížení souborového systému
Zjištění typu souboru
Prohlížení textových souborů
Hledání nápovědy

Přihlášení, aktivace uživatelského rozhraní, odhlášení

Úvod

Abyste mohli s linuxovým systémem přímo pracovat, musíte zadat uživatelské jméno a heslo. Vždy se musíte systému autentizovat. Jak už jsme se zmínili ve cvičení v první kapitole, většina linuxo-vých systémů pro klasické PC může běžet ve dvou základních režimech: rychlém a strohém tex-tovém režimu, který vypadá jako DOS s podporou myši, víceúlohového a víceuživatelského pro-středí, anebo v grafickém režimu, který je pěknější, má ale větší nároky na systémové prostředky.

Grafický režim

Na většině stolních počítačů jde dnes o standardní režim. Grafický režim poznáte velmi snadno už při přihlašování, kdy vám systém nabízí grafické okno pro zadání uživatelského jména a hesla. Chcete-li se přihlásit, přemístěte ukazatel myši do přihlašovacího okna, zadejte své uživatelské jméno a heslo a následně klepněte na OK či zmáčkněte Enter.

> Pozor na účet root!
>
> Obecně je považováno za nevhodné přihlašovat se do grafického prostředí jako uživatel *root*, tedy jako administrátor systému. V grafickém režimu totiž běží celá řada pomocných programů, které by v takovém případě měly zbytečně velká práva. Kvůli minimalizaci rizi-ka se doporučuje přihlašovat se do grafického režimu jako normální uživatel. Tato rada ovšem platí obecně pro jakékoliv použití superuživatelského účtu: Používejte jej pouze tehdy, kde je to nezbytné.

Po zadání uživatelského jména a hesla může chvíli trvat, než dojde ke spuštění grafického pro středí. Závisí to na rychlosti počítače, používaných programech a různých osobních nastaveních. Abyste mohli pokračovat, potřebujete otevřít *terminálové okno*, zkráceně *xterm*. (X Window, zkrá-ceně X, je název softwarového systému, který zajišťuje grafické prostředí.) Tento program nalez-nete v nabídce Aplikace -> Nástroje, Systémové nástroje nebo Internet podle toho, jakou variantu grafického prostředí používáte. Možná váš systém bude obsahovat přímo ikonu, kterou můžete použít jako zkratku pro spuštění terminálu (xterm, rxvt, konsole apod.), a pokud klepnete pra-vým tlačítkem myši na pracovní plochu, objeví se nabídka, která většinou rovněž obsahuje příkaz pro spuštění terminálu.

Zkusíte-li si prohlédnout různé nabídky, nejspíš zjistíte, že celou řadu věcí dokážete velmi snad-no bez toho, aniž byste zadávali nějaké příkazy na klávesnici. Stará dobrá metoda ukaž-a-klepni většině uživatelů postačí ke všemu, co potřebují se systémem udělat. Tato příručka je ovšem urče-na budoucím administrátorům, kteří budou potřebovat měnit ta nejinternější nastavení systému. K tomu budou potřebovat mnohem mocnější nástroj, než je myš. Takovým nástrojem je shell, který v grafickém režimu zpřístupníte právě otevřením terminálového okna.

Okno terminálu představuje ovládací panel systému. Pomocí tohoto jednoduchého, avšak moc-ného textového nástroje je možné udělat prakticky cokoliv. V okně terminálu byste vždy měli vidět výzvu příkazového řádku, takzvaný *prompt*. Následující obrázek ukazuje terminálové okno se standardní výzvou, která obsahuje přihlašovací jméno uživatele a aktuální adresář, reprezentova-ný znakem vlnovky (~):

Další běžný typ výzvy vypadá takto:

[uživatel@systém adresář]

Tato výzva obsahuje jméno přihlášeného uživatele, název počítače, na němž pracuje, a aktuálně otevřený adresář souborového systému.
O výzvách terminálu a jejich chování budeme podrobněji hovořit později. V této chvíli nám stačí vědět, že výzva může obsahovat různé informace, nejsou však součástí příkazů, které budete systému zadávat.

Chcete-li se od systému v grafickém režimu odhlásit, nejprve byste měli zavřít okno terminálu a všechny spuštěné aplikace. Pak klepněte na ikonu odhlášení anebo v nabídce najděte příkaz Odhlásit. V zásadě není nutné ukončovat všechny spuštěné programy, protože systém to udělá za vás, nicméně správce relace možná bude chtít všechny v okamžiku odhlášení spuštěné programy znovu otevřít při příštím přihlášení, což může trvat dlouho a ne vždy je to žádoucí. Toto chování je ovšem možno nastavit.

Jakmile se znovu objeví přihlašovací obrazovka s výzvou k zadání jména a hesla, odhlášení pro-běhlo úspěšně.

Textový režim

Textový režim poznáte podle toho, že je celá obrazovka černá a objevují se na ní (většinou bílé) znaky. Přihlašovací obrazovka v textovém režimu typicky zobrazuje nějaké informace o počítači, na němž pracujete, název počítače a výzvu k přihlášení:

RedHat Linux Release 8.0 (Psyche)

blast login: _

Postup přihlášení se od grafického prostředí poněkud liší. Po zadání uživatelského jména musíte zmáčknout Enter, protože nemáte k dispozici žádná tlačítka ani textová pole, na něž byste mohli klepat myší. Následně zadáte heslo a opět zmáčknete Enter. Při zadávání hesla se nic nevypisuje, dokonce ani hvězdičky, a kurzor se nehýbe. Toto chování je v pořádku a jde o standardní bezpečnostní opatření.

Pokud systém vaše přihlášení přijme, může vypsat nějaké další informace, takzvaný *message of the day*, což může být cokoliv. Na unixových systémech se navíc často vypíše nějaká ta dobrá rada či přísloví. Nakonec se však vždy ocitnete v shellu se stejnou výzvou, jakou znáte z grafického režimu.

> Nepřihlašujte se jako root
>
> Platí to i v textovém režimu: Jako *root* se přihlašujte pouze v případě, že potřebujete pro-vést úlohu, k níž práva administrátora nezbytně potřebujete – například přidat nového uži-vatele, nainstalovat program nebo změnit konfiguraci sítě či jinou konfiguraci systému. Jak-mile nastavení dokončíte, opusťte privilegovaný účet a dále pracujte jako normální uživatel.

Odhlásíte se zapsáním příkazu logout a zmáčknutím klávesy Enter. Dojde k vašemu odpojení od systému a znovu se objeví přihlašovací obrazovka.

> Nevypínejte počítač
>
> Po odhlášení nevypínejte počítač síťovým vypínačem. Počítač byste neměli vypnout, dokud neproběhne standardní procedura zastavení operačního systému. Náhlé vypnutí počítače může vést k různým problémům. Pro tuto chvíli můžete počítač vypnout v grafickém reži-mu tak, že při odhlašování vyberete volbu Vypnout či Zastavit, případně pomocí příslušného tlačítka na ploše či volby v nabídce.

Nyní už se umíme přihlásit a odhlásit, takže můžeme vyzkoušet první příkazy.

Úplné základy

Příkazy

Následující tabulka představuje přehled základních příkazů, které budeme potřebovat. Podrobněji o nich budeme hovořit později.

Příkaz	Význam
ls	Vypíše seznam souborů v aktuálním adresáři, podobně jako příkaz v

	DOSu.
cd *adresář*	Změní aktuální adresář.
passwd	Změní heslo přihlášeného uživatele.
file *název-souboru*	Zobrazí typ souboru se zadaným názvem.
cat *textový-soubor*	Vypíše obsah textového souboru.
pwd	Vypíše název aktuálního (pracovního) adresáře.
exit, logout	Ukončí relaci.
man *příkaz*	Vypíše manuálovou stránku zadaného příkazu.
info *příkaz*	Vypíše informační stránku zadaného příkazu.
apropos *řetězec*	Hledá řetězec v databázi whatis.

Příkazy do začátku

Obecné poznámky

Příkazy zadáváte za výzvou shellu v terminálovém okně grafického režimu nebo v textovém reži
mu, po zapsání příkazu zmáčknete klávesu Enter.Některé příkazy zadáváte samotné, například ls. Chování příkazu můžete ovlivnit různými přepí-nači, které jsou většinou uvedeny pomlčkou (-), například ls -a. Význam konkrétního přepínačemůže být pro různé příkazy různý. GNU programy pracují také s dlouhými přepínači, které se uvo-zují dvěma pomlčkami (--), například ls --all. Některé příkazy nepoužívají žádné přepínače.

Parametr (či parametry) příkazu specifikuje objekt (objekty), na němž se má příkaz provést. Pří-kladem může být příkaz ls /etc, kde je parametrem příkazu ls adresář /etc. Parametrem říkáte, žechcete vidět obsah právě tohoto adresáře, namísto obsahu aktuálního adresáře, což je výchozíchování příkazu ls v případě, že nezadáte žádný parametr. Některé příkazy vyžadují zadání para-metru, u jiných je parametr nepovinný.

Zda a jaké přepínače a parametry příkaz používá, zjistíte pomocí nápovědy k danému příkazu, viz

kapitolu „Hledání nápovědy“.V Linuxu, stejně jako v systému UNIX, se adresáře oddělují normálním lomítkem, stejným, jaké sepoužívá ve webových adresách (v takzvaných URL). Podrobněji budeme o adresářové struktuřehovořit později.
Při práci s adresáři mají znaky . a .. speciální význam. Dozvíme se o něm ve cvičení a v následu

jící kapitole.Vyhýbejte se přihlašování jako administrátor systému, tedy jako *root*. Kromě obvyklých pracov-ních činností lze i většinu jiných úkonů, například kontrolu systému a získání různých informací,provést jako normální uživatel bez speciálních privilegií. Pokud práva administrátora skutečněpotřebujete, například při vytváření nového uživatelského účtu nebo při instalaci programů, dopo-ručeným způsobem je přepnutí uživatelského ID, viz kapitolu „Cesta“.
Téměř všechny příkazy popisované v této příručce lze provést bez práv administrátora. Pokud navíc jako normální uživatel zadáte příkaz či spustíte program, který práva administrátora vyža-duje, systém vás na to upozorní, případně vás rovnou požádá o zadání hesla administrátora. Jak-mile práci s programem doděláte, ihned jej ukončete.

Měli byste se naučit číst dokumentaci. Zejména zpočátku je velmi důležité číst systémovou doku-mentaci, manuálové stránky základních příkazů, dokumenty HOWTO a podobně. Objem dostup-né dokumentace je obrovský, takže není možné uvádět odkazy na všechny související dokumen-ty. V této příručce se budeme snažit upozornit vás u jednotlivých témat na nejvíce relevantní části dokumentace a budeme se snažit vybudovat ve vás zvyk číst manuálové stránky.

Funkce shellu Bash

V GNU shellu Bash můžete pomocí různých klávesových zkratek provádět různé operace snáze a rychleji. Tento shell se nachází prakticky v každém linuxovém systému, viz kapitolu „Shell“. Násle-dující tabulka uvádí přehled běžně používaných funkcí. Doporučujeme vám navyknout si na jejich používání co nejdříve, abyste tak už od začátku mohli využívat výhod, které vám Linux nabízí.

Klávesa nebo kombinace kláves Funkce

Přesune kurzor na začátek příkazového řádku.

Ukončí běžící program a znovu vypíše prompt shellu, viz kapitolu 4, „Procesy“.

Ukončí aktuální relaci, stejně jako příkazy nebo .

Přesune kurzor na konec příkazového řádku.

Smaže znak před kurzorem.

Vymaže obsah terminálového okna.

Prohledává historii příkazů, viz kapitolu o příkazu .

Pozastaví program, viz zmíněnou kapitolu o procesech.

a Přesouvá kurzor na příkazovém řádku o jeden znak vlevo či vpravo, takže můžete dopiso

vat znaky i jinam než jen na začátek či konec řádku. a Prochází historii příkazů. Najděte si příkaz, který chcete zopakovat, případně jej upravte

a zmáčkněte .

a Listuje oknem terminálu. (Můžete procházet text, který už „odroloval“ z obrazovky.)

Dokončení příkazu či názvu souboru. Pokud je k dispozici více dokončení, ozve se obvykle zvu

kový signál, pokud je k dispozici mnoho možností, systém se zeptá, zda je má zobrazit všechny. Ukáže možnosti dokončení příkazu či názvu souboru.

Klávesové zkratky v shellu Bash

Poslední dvě položky v tabulce si možná zasluhují vysvětlení. Pokud se například budete chtít pře-pnout do *adresáře_se_strašlivě_dlouhatánským_názvem*, rozhodně nemusíte celý název takového adresáře vypisovat. Stačí na příkazovém řádku napsat například cd adr a zmáčknout klávesu Tab. Pokud název žádného jiného adresáře nezačíná stejnou trojicí znaků *adr*, shell už celý název adre-sáře doplní. Jestliže neexistuje žádný jiný adresář s názvem začínajícím *a*, stačilo by vám zadat cd a a zmáčknout Tab. Pokud existuje více souborů či adresářů, které začínají stejně, shell vás (obvy-kle zvukovým signálem) upozorní, a pokud zmáčknete dostatečně rychle Tab podruhé, vypíše všechny existující možnosti:

```
výzva> cd st starthere stuff stuffit
```

V tomto případě můžete za první dva znaky *st* dopsat následně například *a*, a jakmile zmáčknete Tab znovu, shell už nemá na výběr a automaticky dokončí název adresáře na *starthere*:

```
výzva> cd starthere
```

Samozřejmě zadání příkazu musíte ještě potvrdit klávesou Enter. Pokud byste ve stejném případě zadali *u* a znovu zmáčkli Tab, shell automaticky doplní znaky *ff* a následně se znovu zarazí, protože má opět na výběr z více variant. Jestliže znovu zmáčknete Tab Tab, objeví se možnosti, můžete dopsat jeden či více znaků, které zajistí jednoznačnost, a znovu můžete zmáčknout Tab, načež shell dokončí zadávání názvu. Po zmáčknutí klávesy
Enter se přepnete do zvoleného adresáře – samozřejmě v případě, že název skutečně je názvem adresáře. Tato metoda funguje při zadávání názvů souborů či adresářů, které slouží jako parametry příka-
zům. Stejným způsobem funguje i doplňování názvů příkazů. Zadáte-li ls a zmáčknete dvakrát klávesu Tab, vypíší se všechny příkazy podle proměnné PATH (viz kapitolu „Cesta“), které začínají touto

dvojicí znaků:

```
výzva> ls ls
lsattr
lsb_release     lsdev     lspci lspgpot     lsraid lss16toppm     lsw
                lsmod lsof  lspnp           lsusb
```

Hledání nápovědy

Upozornění

GNU/Linux se vás snaží vést k samostatnosti. Je obvyklé, že stejnou věc můžete udělat více růz-nými způsoby. Obvyklá metoda, jak se dobrat pomoci, je zeptat se někoho, kdo se v tom vyzná. Jakkoliv může být linuxová komunita trpělivá a mírumilovná, téměř každý předpokládá, že než se zeptáte, sami vyzkoušíte jeden nebo více dále popsaných způsobů. Pokud se tohoto základního pravidla nebudete držet, budou vám je ostatní připomínat způsobem, který nemusí být příliš zdvo-řilý.

Manuálové stránky

Řada začínajících uživatelů má z manuálových stránek obavu, protože jde o velmi obsáhlý zdroj dokumentace. Stránky jsou ovšem velmi dobře strukturované, jak můžete sami zjistit zadáním následujícího příkazu: man man.

Manuálové stránky v grafickém režimu čtete obvykle v okně terminálu, případně rovnou v textovém režimu, pokud vám to více

vyhovuje. Zkuste zadat následující příkaz:

uživatel@počítač ~> man man

Po stisku klávesy Enter se vypíše dokumentace k příkazu man, který slouží k prohlížení manu-álových stránek:

man(1) man(1)

JMÉNO
man - zformátuje a zobrazí on-line manuálové stránky manpath - zobrazí manuálové cesty uživatele

SYNTAXE
man [-acdfhkKtwW] [-m systém] [-p řetězec] [-C konfigurační soubor] [-M cesta] [-P stránkovač] [-S seznam_sekcí] [sekce] jméno ...

POPIS man zformátuje a zobrazí on-line manuálové stránky. Tato verze umí pracovat s proměnnými prostředí MANPATH a (MAN)PAGER, proto můžete mít i vlastní manuálové stránky a vlastní program určený ke stránkování zformátovaných manuálových stránek. Je-li specifikována sekce, man hledá danou stránku pouze v této sekci. Samozřejmě můžete také speci-fikovat pořadí sekcí, které budou prohledávány, a také můžete přímo na příkazové řádce nebo proměnnými prostředí určit, které preprocesory budou při formátování stránek použity. Obsahuje-li jméno znak /, je nejprve vyzkoušeno jako jméno souboru, proto můžete udělat něco jako man ./něco.5 nebo man /cédéčko/něco/něco_jiného.1.gz.

VOLBY -C konfigurační_soubor
Specifikujete jiný konfigurační soubor. Standardní je /etc/man.config. (Viz též man.conf(5).) ...

Stisknutím mezerníku se vypíše další stránka manuálu. Klávesou b se můžete vrátit na předchozí stránku. Jakmile dojdete na konec, prohlížeč se obvykle ukončí a znovu se objeví výzva příkazo-vého řádku. Pokud chcete prohlížení ukončit dříve nebo pokud prohlížeč ve vašem systému neskončí automaticky, ukončíte jej klávesou q.

Stránkovací programy

Klávesové zkratky používané při prohlížení manuálových stránek jsou dány tím, jaký strán-kovací program vaše distribuce používá. Ve většině distribucí se při prohlížení používá pro-gram *less*. O stránkovacích programech se více dozvíte v kapitole „less“.

Každá manuálová stránka obvykle obsahuje několik standardních sekcí, které můžete vidět i v uvedeném příkladu:

První řádek obsahuje název programu, o kterém si právě čtete, a identifikátor sekce manu-álu, ze které příslušná stránka pochází.
Manuálové stránky jsou členěny do kapitol. U pří-kazů bývá obvyklé, že mají svou stránku ve více kapitolách, například v kapitole určené uživatelům, kapitole určené správcům a kapitole určené programátorům
Následují název příkazu a jeho stručný popis, z nichž se generuje rejstřík manuálových strá-nek. Tento rejstřík můžete prohledávat příkazem apropos
V syntaktické části je pomocí technické notace uveden přehled všech přepínačů a/nebo parametrů, kterým příkaz rozumí. Přepínače udávají způsob, jakým má být příkaz spuštěn. Parametry říkají, na co má být příkaz použit. Některé příkazy nemají žádné přepínače ani parametry. Nepovinné přepínače a parametry se uvádějí v hranatých závorkách, „[“ a „]“, což znamená, že je není nutné uvádět
Následuje podrobnější popis příkazu
Dále jsou popsány jednotlivé přepínače a jejich význam. Obvykle je možné různé přepí-nače kombinovat, pokud ne, dozvíte se to v této části
Mohou následovat části specifické pro daný příkaz, u příkazu man je to konkrétně část věnovaná zformátovaným stránkám
V části věnované prostředí jsou popsány proměnné prostředí, jejichž nastavení ovlivní cho-vání příkazu. (Tato část nebývá u všech příkazů.)
V části „Viz též“ naleznete odkazy na další manuálové stránky (v případě anglických manu-álových stránek hledejte „see also“).
V závorkách je uvedeno číslo kapitoly manuálu, v níž se stránka nachází. Zkušenější uživatelé velmi často přecházejí rovnou do této části tak, že zmáčknou klávesu /, zapíší text „viz“ a stisknout Enter. Tímto způsobem můžete v dané stránce hledat obecně jakýkoliv text. Lomítko, hledaný text, Enter nalezne první výskyt textu, klávesou n skočíte na další výskyt
Obvykle následuje část věnovaná známým chybám či anomáliím příkazu a adresa, na kterou můžete hlásit nově objevené chyby
Stránka může končit zmínkou o autorech či autorských právech

Některé příkazy jsou popsány na několika manuálových stránkách. Například příkaz passwd je popsán v části 1 i v části 5. V takových případech příkaz man standardně vypíše stránku ze sekce s nejnižším číslem. Pokud vás zajímá stránka z jiné sekce, uveďte za příkazem man její číslo:

man 5 passwd

Pokud vás zajímají všechny stránky daného příkazu, můžete použít přepínač -a:

man -a passwd

Jakmile dojdete na konec stránky z první sekce a zmáčknete mezerník, objeví se stránka z násle
dující sekce manuálu. V některých distribucích můžete mít část manuálových stránek česky, a to proto, že man dává při vyhledávání stránek přednost nastavenému jazyku. Balíček, který přeložené manuálové stránky obsahuje, se obvykle jmenuje man-pages-cs. Nechcete-li české překlady používat, můžete jej s kli-dem odinstalovat. Aktuální informace o českém překladu manuálových stránek a také aktuální verzi překladu najdete na adrese *http://sweb.cz/tropikhajma/man-pages-cs/index.html.* Na stejném místě se můžete připojit k překladu dalších stránek.

Další informace Informační stránky

Kromě manuálových stránek existují k příkazům i informační stránky, které můžete zobrazit pří-kazem info. Obvykle obsahují novější informace a snáze se používají. Na informační stránky bývá někdy odkazováno i z manuálu.

Můžete si vyzkoušet zadat v terminálovém okně příkaz info info:

Soubor: info.info, Uzel: Nejvyšší, Další: Začínáme, Nahoru: (adr)

Info: Úvod **********

Info je program, který slouží k zobrazení dokumentace k počítačovým programům. Projekt GNU distribuuje většinu on-line manuálů ve formátu „Info“, k jehož čtení potřebujete program zvaný „Info reader“. Jeden takový právě používáte.
Pokud s programem Info pracujete poprvé a chcete vědět, jak se používá, zmáčk-něte „h“. Zobrazí se instrukce pro práci s programem.

Zajímají-li vás pokročilé příkazy, dvakrát zmáčkněte „n“. Zobrazí se informace pro experty, přeskočíte část pro začátečníky.

* Nabídka:

* Začínáme:	Začínáme pracovat s programem Info.
* Pro experty:	Pokročilé příkazy programu Info.
* Vytvoření info souboru:	Jak vytvořit vlastní soubor Info.

--zz-Info: (info.info.gz)Top, 24 lines --Top----------V咤 v疽 Info verze 4.2. C-h zobraz n疳ovědu, m zobraz nab冝ku.

Pomocí šipek se můžete posouvat po textu. Pokud najedete na řádek začínající hvězdičkou, může-te zmáčknout Enter a přesunete se na příslušnou stránku. Klávesami P a N se můžete posouvat k předchozímu a následujícímu tématu. Mezerník vás posune o stránku dál, bez ohledu na to, zda jde o nové téma či o stránku k jinému příkazu. Prohlížení ukončíte klávesou Q. Více informací se dozvíte v programu info.

Příkazy whatis a apropos

Příkazem whatis získáte krátké vysvětlení k příkazu tak, jak to ukazuje následující příklad:

[výzva] whatis ls ls (1) - vypíše obsah adresářů

Zobrazí se stručná informace o příkazu a číslo první manuálové sekce, která příslušnou stránku

obsahuje.Pokud nevíte, kde začít a kterou manuálovou stránku si přečíst, pomůže vám příkaz apropos.Řekněme, že hledáte nějaký prohlížeč (tedy *browser*) – pak můžete zadat následující příkaz:

výzva> apropos browser Galeon [galeon](1) - gecko-based GNOME web browser lynx (1) - a general purpose distributed information browser for the World Wide Web ncftp (1) - Browser program for the File Transfer Protocol opera (1) - a graphical web browser pilot (1) - simple file system browser in the style of the Pine Composer pinfo (1) - curses based lynx-style info browser pinfo [pman] (1) - curses based lynx-style info browser viewres (1x) - graphical class browser for Xt

Po zmáčknutí klávesy Enter získáte seznam různých prohlížečů, které váš systém nabízí. Nejde jen o webové prohlížeče, ale i o prohlížeče souborů, FTP a dokumentace. Pokud máte nainstalo-vány i vývojové balíčky, mohou se objevit i informace o vytváření programů, které mají něco spo-lečného s prohlížením. Obecně platí, že pro normálního uživatele jsou vhodné k vyzkoušení programy popisované v první sekci manuálu, tedy ty příkazy, u nichž je uvedeno „(1)“. Uživatel, který zadal výše uvedený příkaz, tak může následně vyzkoušet příkazy galeon, lynx či opera, což jsou různé webové prohlížeče.

Přepínač - - help

Většina GNU příkazů podporuje přepínač --help, který vypíše stručné informace o použití pro-gramu a přehled možných voleb. Takto vypadá výpis pro příkaz cat:

```
uživatel@systém: cat --helpUsage: cat [OPTION] [FILE]...Concatenate FILE(s), or standard input, to standard output.

  -A, --show-all equivalent to -vET
  -b, --number-nonblank number nonblank output lines
  -e equivalent to -vE
  -E, --show-ends display $ at end of each line
  -n, --number number all output lines
  -s, --squeeze-blank never more than one single blank line
  -t equivalent to -vT
  -T, --show-tabs display TAB characters as ^I
  -u (ignored)
  -v, --show-nonprinting use ^ and M- notation,
                                                    except for LFD and TAB
      --help display this help and exit
      --version output version information and exit

With no FILE, or when FILE is -, read standard input.

Report bugs to <bug-textutils@gnu.org>.
```

Grafická nápověda

Pokud dáváte přednost grafickému rozhraní, nemusíte se obávat. Například Konqueror, výchozí správce souborů v prostředí KDE, nabízí příjemný a barevný přístup k manuálovým a informač-ním stránkám. Do řádku adresy zkuste napsat info:info a objeví se informační stránka příkazu info. Podobně můžete zadat man:ls a zobrazíte manuálovou stránku příkazu ls. Funguje dokon-ce i dokončování příkazů – v rozbalovací nabídce se ukážou všechny příkazy začínající na znaky ls. Pokud v řádku adresy zadáte info:/dir, objeví se seznam všech informačních stránek, rozdě-lený do kategorií. Jde o vynikající zdroj nápovědy. Můžete jej spustit z nabídky nebo v terminálovém okně příkazem konqueror.

Stejně praktický je prohlížeč nápovědy v prostředí Gnome. Můžete jej spustit z nabídky Programy -> Nápověda, pomocí ikony záchranného kruhu anebo přímo příkazem gnome-help v terminá-lovém okně. Přehledné rozhraní vám umožní prohlížet si systémovou dokumentaci i manuálové stránky.

Správce souborů Nautilus umožňuje prohledávat rejstřík manuálových a informačních stránek, které následně můžete zobrazit a využít jejich vzájemného propojení. Nautilus spustíte buď z pří-kazové řádky, klepnutím na ikonu domečku nebo z nabídky Gnome.

Výhodou grafických prohlížečů dokumentace je, že všechny informace jsou vzájemně provázány, takže v části „viz též“ můžete klepnout na zvolený odkaz a přejít rovnou na související manuálovou stránku. Tímto způsobem můžete při čtení dokumentace strávit celé hodiny.

Výjimky

Některé příkazy nemají samostatnou dokumentaci, protože jsou součástí jiného příkazu. Příkladem takových příkazů jsou cd, exit, logout nebo pwd. Tyto příkazy jsou přímo součástí shellu, jde
o takzvané vestavěné příkazy shellu. Informace k těmto příkazům získáte prostřednictvím manu-álových nebo informačních stránek shellu. Většina linuxových systémů nabízí jako výchozí shell Bash. Více informací o shellech se dozvíme v kapitole „Shell“.

Pokud změníte výchozí nastavení systému, může se stát, že manuálové stránky se sice v systému nacházejí, nejsou ale viditelné, protože je změněna příslušná proměnná prostředí. V takovém pří-padě musíte zkontrolovat proměnnou MANPATH. Více se o tomto tématu dozvíte v kapitole „Pro-měnné prostředí“.

Některé programy nebo balíky obsahují pouze nápovědu uloženou v adresáři /usr/share/doc. O prohlížení souborů hovoříme v kapitole „Další způsoby zobrazení obsahu souboru“, která začíná na straně 110.

V nejhorším případě může dojít k tomu, že si dokumentaci ze systému omylem vymažete. (Před-pokládáme, že omylem, protože udělat to úmyslně není vůbec moudré.) V takovém případě nej-prve pomocí nějakého prohledávacího nástroje zkontrolujte, že skutečně nic nezůstalo, viz kapi-tolu „Hledání souborů“. Následně budete muset znovu nainstalovat balíčky s programy, k nimž potřebujete dokumentaci, viz kapitolu „Instalace nových programů“.

Shrnutí

Linux může pracovat v textovém nebo v grafickém režimu. Vzhledem k tomu, že procesorový výkon ani kapacita paměti nejsou dnes cenově nedostupné, může si většina uživatelů dovolit gra-fický režim a obvykle jej také používá. Neznamená to ovšem, že o

textovém režimu nepotřebuje-te nic vědět. V této příručce prostřednictvím terminálového okna používáme i textový režim.

Linux uživatelům umožňuje přístup k informacím a vede je k nezávislosti. K tomu ovšem potře-bujete přečíst spoustu dokumentace, proto si můžete všimnout, že vás téměř u každého příkazu budeme odkazovat na další zdroje informací. Čím více dokumentace přečtete, tím to budete mít jednodušší a tím rychleji se od manuálů odpoutáte. Snažte si na čtení dokumentace co nejvíce zvyknout. Jakmile si s něčím nevíte rady, měli byste automaticky začít studovat dokumentaci.

Cvičení

Většinu věcí se učíme z vlastních chyb, zjištěním toho, co se může pokazit. Následující cvičení vás seznámí s různými chybovými hlášeními. Pořadí, v němž budete cvičení provádět, je důležité. Nezapomínejte také na funkce, které vám Bash při práci s příkazovým řádkem nabízí – snažte se příklady zadat co možná nejmenším počtem stisknutých kláves!

Přihlášení a odhlášení

Zjistěte si, zda pracujete v textovém, nebo grafickém režimuPracuji v textovém/grafickém režimu. (Nehodící se škrtněte.)
Přihlaste se jménem a heslem, které jste si nastavili při instalaci
Odhlaste se
Přihlaste se znovu, použijte neexistující uživatelské jméno
-> Co se stane?

Hesla

Znovu se přihlaste svým jménem a heslem
Změňte si heslo na *P6p3.aa!* a zmáčkněte Enter-> Co se stane?
Změňte heslo znovu, tentokrát na něco velmi jednoduchého, jako *123* nebo *aaa*-> Co se stane?
Zkuste to znovu, tentokrát nezadávejte heslo a jen zmáčkněte Enter-> Co se stane?
Zkuste místo příkazu passwd zadat psswd-> Co se stane?

Nové heslo
Pokud nezměníte heslo na původní hodnotu platnou před tímto cvičením, zůstane vám nastaveno heslo *P6p3.aa!*. Po skončení cvičení si nezapomeňte heslo změnit!
Některé systémy nepovolují recyklovat hesla, není tedy možné vrátit heslo zpět na již jed nou použitou hodnotu, pokud neuplyne nějaký čas nebo heslo nebylo vícekrát změněno.

Adresáře

Následuje několik příkladů, abyste získali představu o adresářích
Zadejte příkaz cd blah-> Co se stane?
Zadejte příkaz cd(Všimněte si mezery mezi cd a tečkami!) Zadejte příkaz pwd-> Co se stane?
Vypište obsah adresáře příkazem ls-> Co vidíte?-> Co si myslíte, že to znamená?-> Vyzkoušejte příkaz pwd
Zadejte příkaz cd-> Co se stane?
Zadejte dvakrát příkaz cd-> Co se stane?
Vypište obsah tohoto adresáře
Vyzkoušejte příkaz cd root-> Co se stane?-> Do kterých adresářů máte přístup?
Zadejte příkaz cdJak jinak byste se mohli dostat do tohoto místa?

Soubory

■ Změňte adresář na / a následně na etc. Zadejte příkaz ls, pokud je výstup delší než veli-kost terminálového okna, zvětšete okno nebo vyzkoušejte klávesy Shift+PageUp a Shift+PageDown
Soubor inittab obsahuje odpověď na první otázku z tohoto cvičení. Zkuste na něj spus-tit příkaz file -> Soubor inittab je typu...
Použijte příkaz cat inittab a přečtěte si obsah souboru-> Jaký je výchozí režim spuštění vašeho počítače?
Příkazem cd se vraťte do svého domovského adresáře
Zadejte příkaz file-> Zjistili jste, co znamená „.“?
Můžete si „.“ prohlédnout příkazem cat?
Pomocí přepínače --help si vypište nápovědu k příkazu cat. Použijte přepínač pro číslo-vání řádků a zjistěte tak počet řádků v souboru /etc/passwd

Nápověda

Přečtěte si man intro
Přečtěte si man ls

Přečtěte si info passwd
Zadejte příkaz apropos pwd
Zkuste man nebo info na příkaz cd -> Jak se dozvíte podrobnosti o příkazu cd?
Přečtěte si ls --help a vyzkoušejte různé přepínače

Soubory a souborové systémy

Po prvním seznámení, které jsme provedli ve druhé kapitole, se nyní můžeme věnovat souborům a adresářům v linuxovém systému podrobněji. Řada uživatelů má s Linuxem problém z toho důvo-du, že nemají přehled o tom, jaká data jsou kde umístěna. Pokusíme se proto poněkud objasnit umístění souborů v souborovém systému.
Seznámíme se také s nejdůležitějšími soubory a adresáři a ukážeme si různé způsoby prohlížení souborů. Předvedeme si rovněž, jak soubory a adresáře vytvářet, přesouvat a mazat. Po provedení cvičení v této kapitole byste měli být schopni:

Popsat strukturu linuxového souborového systému
Zjistit nastavení cesty pro spouštění programů
Popsat nejdůležitější soubory včetně jádra a shellu
Nalézt ztracené a skryté soubory
Vytvářet, přesouvat a mazat soubory a adresáře
Zobrazit obsah souborů
Chápat význam a užití různých typů odkazů
Zjistit vlastnosti souboru a změnit práva souboru

Obecný přehled systému souborů

Soubory Obecný úvod

Jednoduchá definice ze systému UNIX, která platí také pro Linux, říká:V unixovém systému je všechno soubor. Pokud něco není soubor, je to proces.Toto tvrzení je pravdivé, protože existují speciální soubory, které jsou mnohem víc než jenom soubory (například pojmenované roury nebo sokety). Abychom to však příliš nekomplikovali, rozum-né zobecnění říká, že všechno je soubor. Linux, stejně jako UNIX, nerozlišuje mezi souborema adresářem, protože adresář je pouze soubor, který obsahuje názvy jiných souborů. Programy,služby, texty, obrázky a podobně, všechno je uloženo jako soubory. I vstupní a výstupní zařízení, obecně jakékoliv zařízení, systém chápe jako soubory.
Aby bylo možné se všemi těmito soubory rozumně pracovat, rádi si je představujeme umístěné na disku ve stromově organizované struktuře, kterou známe například z dob MS-DOS. Hlavní větve obsahují další větve, koncové větve pak obsahují listy stromu, tedy obyčejné soubory. Pro tuto chvíli budeme pracovat s takovouto stromovou představou, později však uvidíme, že taková představa není úplně přesná.

Typy souborů

Většina souborů jsou prostě soubory, normální soubory, které obsahují normální data. Například textové soubory nebo spustitelné soubory (či programy), soubory se vstupními či výstupními daty různých programů a podobně.

I když ve většině případů můžete poměrně neškodně předpokládat, že cokoliv, na co v linuxo-vém systému narazíte, je soubor, existují i výjimky.

Adresáře: Soubory, které obsahují seznam jiných souborů.
Speciální soubory: Představují mechanismus vstupu a výstupu. Většinu speciálních soubo-rů naleznete v adresáři /dev, budeme o nich hovořit později
Odkazy: Mechanismus, který umožňuje zpřístupnit soubor či adresář na více místech sou-borového stromu. O odkazech budeme podrobněji hovořit
(Doménové) sokety: Speciální typ souborů, podobný soketům protokolu TCP/IP, který slou-ží jako prostředek komunikace mezi procesy podléhající ochranným mechanismům sou-borového systému
Pojmenované roury: Fungují víceméně podobně jako sokety a rovněž umožňují vzájemnou komunikaci mezi procesy, tentokrát bez nutnosti použít sémantiku síťových soket

Přepínač -l příkazu ls vypisuje typ souborů, poznáte jej podle prvního znaku na každém řádku:

jaime:~/Documents> ls -l total 80 -rw-rw-r--1 jaime jaime 31744 Feb 21 17:56 intro Linux.doc -rw-rw-r--1 jaime jaime 41472 Feb 21 17:56 Linux.doc drwxrwxr-x 2 jaime jaime 4096 Feb 25 11:50 course

Význam jednotlivých znaků shrnuje následující tabulka:

Znak	Význam
-	normální soubor
d	adresář
l	odkaz
c	znakové zařízení (speciální soubor)
s	soket
p	pojmenovaná roura
b	blokové zařízení (speciální soubor)

Typy souborů

Abyste nemuseli typy souborů zjišťovat pokaždé pomocí dlouhého výpisu, většina systémů standardně spouští nikoliv příkaz ls, nýbrž ls -F, který znázorňuje typ souboru jedním ze znaků /=*|@, připojeným za název souboru. Aby měli začínající uživatelé ještě jednodušší život, ve většině pří-padů se standardně kombinují parametry -F a --color, viz kapitolu „Zobrazení vlastností soubo-
ru“. Kvůli zlepšení čitelnosti používáme i v této příručce standardně příkaz ls -F.Jako uživatel přijdete do přímého styku pouze s normálními soubory, spustitelnými soubory, adre-sáři a odkazy. Další speciální typy souborů slouží k tomu, aby váš systém mohl dělat to, co od nějpotřebujete, a manipulace s těmito soubory je starost správce systému a programátora.

Než si uvedeme přehled důležitých souborů a adresářů, potřebujeme se ještě dozvědět něco o diskových oddílech.

O diskových oddílech K čemu diskové oddíly?

Většina uživatelů má o existenci diskových oddílů (či *particí*) jen letmou představu, protože každý operační systém dokáže oddíly vytvářet a rušit sám. Může vypadat podivně, že Linux i při standardní instalaci vytvoří na jednom disku více diskových oddílů, takže bude vhodné si to nějak vysvětlit.

Jedna z výhod při použití více oddílů spočívá ve větší bezpečnosti dat v případě havárie. Pokud disk rozdělíme na oddíly, můžeme data rozdělit do skupin a vzájemně oddělit. Dojde-li k havárii, ztratí se pouze data na postiženém oddíle, zatímco data na ostatních oddílech zůstanou obvykle nepoškozená.

Tento princip pochází z doby, kdy Linux nepoužíval žurnálovací souborové systémy a výpadek napájení mohl vést ke katastrofě. Diskové oddíly se i nadále používají kvůli bezpečnosti a robust-nosti, takže problém v jedné části systému nemusí nutně znamenat ohrožení celého počítače. Dnes je to v zásadě hlavní důvod pro použití samostatných oddílů. Jednoduchý příklad: Uživatel vytvoří skript či program, který začne trvale zapisovat na disk. Pokud by byl celý systém nainsta-lován na jediném oddílu, došlo by k jeho zaplnění a systém by se zastavil. Pokud se ovšem uži-vateli vytvářená data ukládají na samostatný oddíl, zaplní se pouze tento oddíl a ostatní oddíly, například systémový, budou i nadále funkční.

Žurnálovací souborový systém zabezpečuje data pouze před nečekaným výpadkem napájení nebo odpojením diskového zařízení. Neochrání vás před chybami na disku a logickými chybami sou-borového systému. Potřebujete-li takový typ ochrany, můžete použít nějaké řešení založené na technice RAID (Redundant Array of Inexpensive Disks).

Typy a rozdělení diskových oddílů

V linuxovém systému existují dva hlavní typy diskových oddílů:

Datový oddíl: běžný diskový oddíl pro ukládání dat, jedním z nich bývá oddíl *root* obsa-hující data potřebná pro spuštění a běh systému
Odkládací oddíl: rozšíření fyzické paměti počítače, dodatečná paměť na pevném disku

Většina systémů obsahuje kořenový oddíl, jeden nebo více datových oddílů a jeden nebo více odkládacích oddílů. Systémy v heterogenním prostředí mohou obsahovat oddíly pro data jiných systémů, například oddíly se souborovými systémy FAT nebo VFAT pro data operačního systému MS Windows.

K vytvoření diskových oddílů při instalaci používá většina linuxových systémů příkaz fdisk. Jak jste asi postřehli ve cvičení z první kapitoly, k rozdělení disku obvykle dojde automaticky. Ne vždy ale budete mít takové štěstí. V takových případech budete muset nastavit typ oddílu ručně a ručně disk na oddíly rozdělit. Standardní linuxové diskové oddíly mají typ 82 (odkládací) a 83 (datový), který může být žurnálovací (ext3) nebo normální (ext2, na starších systémech). Pokud byste si tyto hodnoty nezapamatovali, obsahuje příkaz fdisk vestavěnou nápovědu.
Kromě již zmíněných souborových systémů ext2 a ext3 Linux podporuje množství dalších soubo-rových systémů, například ReiserFS, JFS, NFS, FAT*xx* a celou řadu dalších, používaných nativně na jiných (proprietárních) operačních systémech.

Kořenový diskový oddíl (označovaný jedním lomítkem, */*) mívá velikost 100 až 500 MB a obsahuje konfigurační soubory systému, základní příkazy a služební programy, systémové knihovny, odklá-dací prostor a domovský adresář administrátora. Při standardní instalaci je vyžadována velikost alespoň 250 MB. Instalace pracovní stanice s jednoduchým rozdělením disku, zmíněná dále (pouze / a /home), bude pravděpodobně vyžadovat kořenový oddíl v řádech gigabajtů.

Odkládací oddíl je přístupný pouze samotnému operačnímu systému a za normálních okolností je skrytý. Systém odkládání paměti na disk v Linuxu stejně jako v jiných UNIXech zajišťuje, že bez ohledu na situaci můžete pokračovat v práci. V Linuxu se vám nestane, že by se objevilo chybo-vé hlášení „Nedostatek paměti, ukončete nějaké programy a zkuste to znovu“, právě proto, že odkládací oddíl vám nabízí paměť navíc. Tento mechanismus virtuální paměti je dnes součástí všech moderních operačních systémů i mimo unixový svět.

Používat paměť na pevném disku je samozřejmě pomalejší než práce s „opravdovou“ pamětí, při-náší to však s sebou nezanedbatelné pohodlí. O virtuální paměti budeme hovořit ve čtvrté kapi-tole v části věnované procesům.

Typicky se doporučuje instalovat Linux tak, aby velikost odkládacího oddílu byla dvojnásobkem velikosti fyzické paměti počítače. Při instalaci se musíte rozhodnout, jak to udělat. Máte-li například 512 MB fyzické paměti, můžete si vybrat mezi několika variantami:

jeden odkládací oddíl o velikosti 1 GB,
dva odkládací oddíly o velikosti 512 MB,
máte-li dva disky, odkládací oddíl o velikosti 512 MB na každém disku,

Poslední varianta dává nejlepší výsledky, pokud se u systému předpokládá velké množství vstup
ně-výstupních operací. Konkrétní podrobnosti se dočtete v dokumentaci. Některé aplikace, například databáze, mají větší nároky na velikost odkládacího prostoru. U jiných systémů nemusí být odkládací prostor vytvořen vůbec, například proto, že nepoužívají pevný disk. V případě přenosných počítačů (notebooků) nastavte odkládací prostor minimálně dvakrát tak velký jako paměť RAM, jinak byste mohli mít problémy s uspáváním systému na disk. Nastavení odkládacího oddílu se dále může lišit i podle používaného jádra.

Ve většině distribucí bývá jádro umístěno na samostatném diskovém oddílu, protože jde o nejdů-ležitější komponentu systému. V takovém případě je najdete na oddílu */boot*, kde se nachází jádro a související datové soubory.

Zbytek disku či disků bývá rozdělen na datové oddíly, přičemž není výjimkou, že všechna zbývající data bývají umístěna na jediném oddílu – například pokud instalujete standardní pracovní stanici. Pokud data rozdělujete na samostatné oddíly, obvykle se dodržuje následující dělení:

- oddíl pro uživatelské programy (*/usr*),
- oddíl pro domovské adresáře uživatelů (*/home*),
- oddíl pro uložení dočasných dat, jako jsou tiskové a poštovní fronty (*/var*),
- oddíl pro programy třetích stran (*/opt*),

Jakmile oddíly jednou vytvoříte, můžete už pouze přidávat další (máte-li kam). Změna velikosti nebo vlastností existujících oddílů je sice možná, ale nedoporučuje se.
Rozdělení disků na jednotlivé oddíly je úkolem správce systému. Na velkých systémech může dokonce pomocí příslušných programů rozdělit jeden diskový oddíl na více disků. Většina distri-bucí nabízí standardní instalaci optimalizovanou pro pracovní stanice (průměrného uživatele) a běžné servery, umožňuje však i vlastní rozdělení diskových oddílů. Při instalaci můžete rozdě-lení diskových oddílů definovat buď specifickým nástrojem dané distribuce (což je obvykle jed-noduché grafické rozhraní) anebo přímo programem fdisk, což je standardní textový nástroj pro vytváření oddílů a nastavení jejich vlastností.

Typickou pracovní stanici obvykle používá stále stejný uživatel. Tomu odpovídá volba programů při instalaci a důraz kladený na všelijaké balíčky, jako jsou témata vzhledu plochy, vývojové nástro-je, poštovní programy, multimediální programy, webové prohlížeče a podobně. Vše bývá nainsta-lováno na jednom velkém oddílu, přidá se odkládací oddíl o velikosti odpovídající dvojnásobku paměti a obecná pracovní stanice je hotová. K osobnímu užití nabízí maximum diskového prosto-ru, nevýhodou ovšem zůstává možné porušení dat v případě vzniku problémových situací.

Na serverech bývá zvykem oddělit systémová data od uživatelských dat. Programy zajišťující potřebné služby bývají umístěny odděleně od dat, s nimiž tyto služby pracují. Na takových systémech bývá vytvořeno více diskových oddílů:

- oddíl s daty potřebnými ke spuštění systému,
- oddíl s konfiguračními údaji a služebními programy,
- jeden nebo více oddílů s daty jednotlivých služeb, například databázovými tabulkami, uživatelskou poštou, ftp archivem a podobně,
- oddíl s uživatelskými programy a aplikacemi,
- jeden nebo více oddílů pro uživatelské soubory (domovské adresáře),
- jeden nebo více odkládacích oddílů (virtuální paměť).

Servery mají typicky více paměti, a tedy i větší odkládací prostor. Některé služby, například data-báze, potřebují mnohem více odkládacího prostoru, než je obvyklé. Podrobnosti naleznete v dokumentaci k takovýmto programům. Kvůli zvýšení výkonů se odkládací prostor často rozdě-luje do několika samostatných oddílů.

Přípojné body

Všechny diskové oddíly se k systému připojují prostřednictvím přípojných bodů. Přípojný bod definuje místo, kde má být v souborovém systému umístěna příslušná část dat. Typicky se jednot-livé oddíly připojují ke kořenovému oddílu. V kořenovém oddílu, označeném lomítkem (/), jsou vytvořeny adresáře. Tyto prázdné adresáře budou sloužit jako vstupní místo na diskové oddíly, které jsou k nim připojeny. Příklad: Představte si oddíl, který obsahuje následující adresáře:

```
videos/ cd-images/ pictures/
```

Tento diskový oddíl budeme chtít do souborového systému připojit v adresáři /opt/media. Aby to bylo možné, musí nejprve adresář /opt/media existovat. Vhodné je, aby byl adresář prázdný. Jak to zařídit, to si ukážeme později. Následně může administrátor pomocí příkazu mount zajistit při-pojení oddílu k souborovému systému. Vypíšete-li si obsah původně prázdného adresáře /opt/media, zjistíte, že nyní obsahuje soubory a adresáře na připojeném médiu (což může být pevný disk, oddíl na disku, CD, DVD, flashdisk, USB nebo jiné zařízení diskového typu).

Při startu systému dojde k automatickému připojení všech oddílů, které jsou uvedeny v souboru /etc/fstab. Některé oddíly se automaticky nepřipojují, například tehdy, pokud nejsou trvalou sou-částí systému – třeba digitální fotoaparát. Pokud je vše správně nastaveno, diskový prostor zaří-zení se automaticky připojí do souborového systému ihned, jakmile systém zjistí, že došlo k fyzic-kému připojení zařízení. Je také možno nastavit souborový systém tak, aby jej mohl připojit i nor-mální uživatel – k připojení a odpojení tak nejsou nutná práva administrátora. Příklad naleznete v kapitole „Zálohování na/z mechaniky Jazz, USB zařízení a podobně“.

Na běžícím systému můžete informace o diskových oddílech a jejich přípojných bodech zjistit pří-kazem df (což je zkratka od *disk full* anebo *disk free*, vyberte si). Linux používá GNU verzi pří-kazu df, která umí pracovat s přepínačem -h (*human readable*), jímž významně zvýšíte čitelnost výstupu. Komerční unixové systémy mívají obvykle vlastní verzi příkazu df i mnoha dalších pří-kazů. Jejich chování je typicky stejné jako u GNU verzí, nicméně GNU verze obvykle nabízejí více různých voleb a vylepšení.

Příkaz df vypisuje informace pouze o aktivních oddílech jiného než odkládacího typu. Může jít i o oddíly síťových služeb, jak to ukazuje následující příklad, kde se domovské adresáře připojují ze souborového serveru přes síť. Takové uspořádání se často používá ve firemním prostředí.

```
freddy:~> df -h
```

Filesystem	Size	Used	Avail	Use% Mounted on
/dev/hda8	496M	183M	288M	39% /
/dev/hda1	124M	8.4M	109M	8% /boot
/dev/hda5	19G	15G	2.7G	85% /opt
/dev/hda6	7.0G	5.4G	1.2G	81% /usr
/dev/hda7	3.7G	2.7G	867M	77% /var
fs1:/home	8.9G	3.7G	4.7G	44% /.automount/fs1/root/home

Více o rozvržení souborového systému Vizualizace

Pro jednoduchost se o linuxovém souborovém systému často hovoří jako o stromové struktuře. Ve standardním linuxovém systému bude rozvržení obecně vypadat tak, jak to ukazuje obrázek na následující straně.

Uvedený obrázek odpovídá organizaci souborového systému v distribuci RedHat. V závislosti na rozhodnutí administrátora systému, použité distribuci a účelu systému může struktura na jiném počítači vypadat jinak, některé adresáře mohou chybět, jiné mohou být navíc. Dokonce ani názvy adresářů nejsou závazné, jde pouze o obvyklou konvenci. Doporučení ohledně adresářové struk-tury v Linuxu formuluje projekt Linux Standard Base, viz *http://lsb.freestandards.org/*.

Strom souborového systému začíná vrcholem, který se označuje lomítkem (/). Tento adresář, který obsahuje všechny vnořené adresáře a soubory, se často označuje jako kořenový adresář či kořen souborového systému.

Názvy adresářů bezprostředně pod kořenovým adresářem se často uvádějí i s úvodním lomítkem, aby se tak naznačilo jejich umístění a předešlo se případné záměně se stejnojmennými adresáři na jiné úrovni. Začínáte-li používat nový systém, vždycky je rozumné seznámit se se strukturou kořenového adresáře. Podívejme se, co tam najdeme:

emmy:~> cd /emmy:/> lsbin/ dev/ home/ lib/ misc/ opt/ root/ tmp/ var/boot/ etc/ initrd/ lost+found/ mnt/ proc/ sbin/ usr/

Adresář	Obsah
/bin	Běžné programy, používané systémem, administrátorem i uživateli.
/boot	Spouštěcí soubory a jádro systému, soubor vmlinuz. V novějších distribucích také data spouštěče . Zkratka GRUB znamená GRand Unified Boot-loader a jde o pokus zbavit se spousty různých zavaděčů, které se dnes používají.
/dev	Obsahuje odkazy na veškerá periferní zařízení počítače, která jsou zde reprezentována soubory s různými speciálními vlastnostmi.
/etc	V tomto adresáři se nacházejí nejdůležitější konfigurační údaje systému, obsahuje všechny údaje, které například ve Windows naleznete v Ovládacích panelech.
/home	Domovské adresáře normálních uživatelů.
/initrd	(Jen v některých distribucích.) Obsahuje informace potřebné pro spuštění systému. Nemazat!
/lib	Knihovny, tedy soubory používané všemi možnými programy, a už uživatelskými nebo systémovými.

Adresář	Obsah
/lost+found	Tento adresář se nachází na každém diskovém oddílu. Ukládají se do něj data zrestaurovaná při obnově disku po nekorektním odpojení.
/misc	Různé.
/mnt	Standardní přípojné místo externích souborových systémů, například CD mechanik a digitálních fotoapa rátů.
/net	Standardní přípojné místo pro síové souborové systémy.
/opt	Obvykle obsahuje dodatečně instalované programy třetích stran.
/proc	Virtuální souborový systém obsahující informace o systému. Podrobnější informace o souborech v tomto adresáři získáte příkazem . Obecné informace o tomto souborovém systému naleznete v souboru proc.txt (je v adresáři s dokumentací dodávanou ke zdrojovému kódu jádra).
/root	Domovský adresář administrátora. Nezaměňujte adresář / (kořenový adresář) s adresářem /root, domov- ským adresářem uživatele root (stejně se čtou).
/sbin	Programy používané systémem a administrátorem.
/tmp	Dočasný odkládací prostor používaný různými programy, při startu systému se maže. Neukládejte sem proto žádná svá data!
/usr	Programy, knihovny, dokumentace a podobně ke všem uživatelským programům.
/var	Místo pro ukládání proměnných a dočasných souborů, jako jsou různé záznamy, fronty, soubory stahova né z Internetu a podobně.

Podadresáře v kořenovém adresáři

Jak zjistíte, na kterém diskovém oddílu se nachází konkrétní adresář? Spusťte příkaz df a jako para-metr zadejte tečku (.). Příkaz v takovém případě vypíše pouze informace o oddílu, na němž se nachází aktuální adresář:

sandra:/lib> df -h .Filesystem Size Used Avail Use% Mounted on/dev/hda7 980M 163M 767M 18% /

Obvykle platí, že všechny adresáře přímo pod kořenovým adresářem jsou na kořenovém svazku. Pokud ne, pak v úplném výpisu příkazu df (nebo df -h, bez dalších parametrů) najdete pro daný adresář samostatný záznam.

Další podrobnosti můžete zjistit příkazem man hier.

Reálný pohled

Pro většinu uživatelů a při většině obvyklých administrativních činností je představa o stromové organizaci souborového systému v pořádku. Operační systém samotný nicméně o stromech nebo stromových strukturách nic neví.

Každý diskový oddíl má svůj vlastní souborový systém. Představíte-li si všechny tyto souborové systémy pohromadě, může to vypadat jako stromová struktura, není to ale tak jednoduché. V sou-borovém systému je každý soubor reprezentován svým *inode*, což je jakési sériové číslo, které obsahuje informace o vlastních datech souboru: Komu soubor patří a kde na disku se nachází jeho obsah.

Každý oddíl používá vlastní mechanismus číslování inodů, v rámci jednoho systému mohou mít inode na různých oddílech stejná čísla.
Každý inode popisuje určitou datovou strukturu na disku. Ta obsahuje různé informace o vlast-nostech souboru včetně toho, kde jsou fyzicky umístěna data souboru. Při inicializaci disku jako datového úložiště, tedy obvykle při instalaci systému nebo při přidání nového disku, se na disko-vém oddílu vytvoří pevně daný počet inodů. Toto číslo představuje maximální počet souborů všech typů (tedy včetně adresářů, speciálních souborů, odkazů a podobně), které mohou na daném diskovém oddílu vzniknout. Typicky se počítá s jedním inode na každých 2 až 8 kilobaj-tů diskového prostoru.

Vytváříte-li nový soubor, přidělí se mu volný inode. V tomto inode jsou pak uloženy následující informace:

Vlastník a skupina vlastníka souboru
Typ souboru (normální, adresář, ...)
Přístupová práva (viz kapitolu „Přístupová práva: primární obranná linie“)
Datum a čas vytvoření, posledního čtení a změny
Datum a čas poslední změny údajů v inode
Počet odkazů na daný soubor (viz dále v této kapitole)
Velikost souboru
Adresa definující umístění vlastních dat souboru

Mezi údaji uloženými v inode chybí název souboru a název adresáře. Tyto údaje jsou uloženy ve speciálních adresářových souborech. Prostřednictvím kombinace názvu souboru a čísla inode může systém vytvořit stromovou reprezentaci souborového systému, která je názorná pro uživa-tele. Čísla inodů můžete vypsat přepínačem -i příkazu ls. Inody jsou uloženy ve své vlastní oblas-ti diskového oddílu.

Orientace v souborovém systému

Cesta

Když chcete spustit nějaký příkaz, téměř nikdy nemusíte uvádět plnou cestu k souboru, v němž je příkaz uložen. Víme například, že příkaz ls je uložen v adresáři /bin (můžete to zjistit příkazem which -a ls), pokud ale chceme vypsat obsah nějakého adresáře, nemusíme zadávat příkaz /bin/ls.

O tyto věci se stará proměnná prostředí PATH. Tato proměnná obsahuje seznam adresářů, v nichž se nacházejí spustitelné soubory, a šetří tak uživateli práci s pamatováním a zadáváním úplných cest k příkazům. Tato proměnná proto logicky obsahuje spoustu adresářů, v jejichž názvu se někde vyskytuje řetězec bin tak, jak to ukazuje následující příklad. Příkazem echo zobrazíme obsah („$“) proměnné PATH:

```
rogier:> echo $PATH /opt/local/bin:/usr/X11R6/bin:/usr/bin:/usr/sbin/:/bin
```

V tomto případě se bude požadovaný program postupně hledat v adresářích /opt/local/bin, /usr/X11R6/bin, /usr/bin, /usr/sbin a /bin. Jakmile bude program v některém adresáři nalezen, spustí se a prohledávání už dále nepokračuje. To může vést k podivným situacím. V prvním následujícím příkladu jeden uživatel ví, že v systému existuje program sendsms slouží k ode-slání SMS, druhý uživatel jej však nemůže spustit. Rozdíl je právě v nastavení cesty obou uži-vatelů:

```
[jenny@blob jenny]$ sendsms bash: sendsms: command not found [jenny@blob jenny]$ echo
$PATH /bin:/usr/bin:/usr/bin/X11:/usr/X11R6/bin:/home/jenny/bin [jenny@blob jenny]$ su - tony Password: tony:~>which sendsms sendsms
is /usr/local/bin/sendsms

tony:~>echo
$PATH /home/tony/bin.Linux:/home/tony/bin:/usr/local/bin:/usr/local/sbin:\ /usr/X11R6/bin:/usr/bin:/usr/sbin:/bin:/sbin
```

Všimněte si použitého příkazu su (*switch user*), který umožňuje spustit shell s jinou uživatelskou identitou, samozřejmě za předpokladu, že znáte heslo onoho druhého uživatele.
Zpětné lomítko symbolizuje pokračování textu na následujícím řádku, text není rozdělen znakem nového řádku.

V následujícím příkladu chce uživatel spustit příkaz wc (*word count*) a spočítat jím počet řádkův souboru, nic se ale nestane a on operaci po chvíli přeruší stiskem kláves Ctrl+C:

```
jumper:~> wc -l test

(Ctrl-C)jumper:~> which wcwc is hashed (/home/jumper/bin/wc)

jumper:~> echo $PATH/home/jumper/bin:/usr/local/bin:/usr/local/sbin:/usr/X11R6/bin:\/usr/bin:/usr/sbin:/bin:/sbin
```

Z výsledku příkazu which vidíme, že uživatel má ve svém vlastním domovském adresáři adresář bin, v němž má rovněž uložen program pojmenovaný wc. Vzhledem k tomu, že v cestě je nejpr-ve uveden podadresář bin domovského adresáře, bude při hledání programu wc dříve nalezen a spuštěn tento „domácí“ program, který pravděpodobně nezvládne zpracovat vstupní soubor, a uživatel jej proto musel násilně ukončit. Popsaný problém lze vyřešit několika způsoby (v Unixu/Linuxu lze každý problém řešit několika způsoby): Jedna možnost je přejmenovat pro-gram wc v adresáři uživatele, druhá možnost je uvést úplnou cestu k tomu programu wc, který uživatel hodlá spustit. Cesty ke všem stejnojmenným programům je možno zjistit příkazem which s přepínačem -a:

```
jumper:~> /usr/bin/wc -l test 10 test
```

Pokud uživatel spouští program wc „v tom druhém“ adresáři častěji, může cestu upravit tak, aby se jeho domovský adresář prohledával až nakonec:

```
jumper:~> export PATH=/usr/local/bin:/usr/local/sbin:/usr/X11R6/bin:\ /usr/bin:/usr/sbin:/bin:/sbin:/home/jumper/bin
```

Tato změna není trvalá

Použijete-li v shellu příkaz export, provedené změny jsou pouze dočasné a platí jen v rámci existující relace (tedy do odhlášení). Otevřete-li novou relaci v době, kdy stávající relace trvá, nebudou v ní provedené změny platit. V kapitole „Textové prostředí“ si uká žeme, jak nastavit změny cesty trvale tak, že ji nastavíme přímo v konfiguračních soubo rech shellu.

Absolutní a relativní cesty

Cesta, tedy adresářová hierarchie, kterou je nutno projít pro nalezení daného souboru, může být specifikována vzhledem k počátku adresářového stromu (od kořenového adresáře). V takovém případě začíná cesta lomítkem a říká se jí absolutní cesta, protože její zadání je zcela jednoznačné – může mu vyhovovat pouze jediný soubor.

V ostatních případech cesta nezačíná lomítkem a může dojít k záměnám – jako u programů ~/bin/wc (v domovském adresáři uživatele) a bin/wc v adresáři /usr tak, jak jsme to viděli v předchozím příkladu. Cesty, které nezačínají lomítkem, jsou vždy relativní.

V rámci relativní cesty můžeme používat i symboly . a .., které zastupují aktuální a rodičovský adresář. Několik praktických příkladů:

Překládáte-li program ze zdrojových kódů, v instalační dokumentaci se často říká, že máte spustit program ./configure, tedy program umístěný v aktuálním adresáři (adresáři s nově překládaným programem), a nikoliv stejnojmenný program umístěný někde jinde v systé-mu
V HTML souborech se relativní cesty používají velmi často proto, aby bylo možné stránky snadno přemístit na jiné místo:

```
<img alt="Garden with trees" src="../images/garden.jpg">
```

- Ještě jednou si všimněte následujícího rozdílu:

```
theo:~> ls /mp3
ls: /mp3: No such file or directory
theo:~>ls mp3/
oriental/ pop/ sixties/
```

Nejdůležitější soubory a adresáře Jádro

Jádro je srdcem systému. Řídí komunikaci mezi hardwarem a jednotlivými periferiemi. Zajišťuje také, že se procesy a démoni (procesy serverů) spouštějí a zastavují v přesně správnou dobu. Jádro má ještě celou řadu jiných důležitých úkolů, tak mnoho, že dokonce existuje specializovaná poš-tovní konference věnovaná jen vývoji jádra, v níž se probírá obrovské množství informací. Podrob-nější debata o jádře je daleko mimo záběr této příručky. Pro tuto chvíli nám stačí vědět, že sou-bor s jádrem je nejdůležitějším souborem v systému.

Shell

Co je to shell?

Najít vhodnou definici shellu se ukázalo jako velmi náročný úkol. Existuje celá řada různých defi-nic, od jednoduchého přirovnání „shell je něco jako volant u auta“, přes vágní definici v manu-álu bashe, která říká, že „bash je příkazový interpretr kompatibilní s *sh*“, až po zcela obskurní vyjádření jako „shell řídí interakci mezi systémem a jeho uživateli“. Shell je ale mnohem více než
jenom to. Nejlépe je možno shell charakterizovat způsobem, jakým uživatel komunikuje s počítačem, komu-nikačním jazykem. Většina uživatelů zná jiný komunikační jazyk – ukaž-a-klikni na ploše. V tomto jazyce ovšem směr konverzace určuje počítač a uživatel má pouze pasivní úkol vybírat si z nabí-zených variant. Pro programátory je velmi obtížné zahrnout do grafického prostředí všechny pře-pínače, volby a parametry, které GNU příkazy používají. Grafické rozhraní je tak ve svých mož-nostech téměř vždy chudší než příkaz či příkazy, které tvoří jeho funkční pozadí.

Oproti tomu shell představuje pokročilou metodu komunikace se systémem, protože umožňuje obousměrnou konverzaci a přebírání iniciativy. Oba účastníci komunikace jsou si rovni, a je tedy možno snadno zkoušet nové nápady. Shell umožňuje uživateli velmi pružný způsob práce s počí-tačem. Jednou z jeho dalších výhod je, že umožňuje různé úkony automatizovat.

Druhy shellů

Stejně jako lidé používají různé jazyky a dialekty, i počítače používají různé typy shellů:

- sh neboli Bourne shell: původní shell používaný v unixových systémech a prostředích. Jde o základní shell, krátký program s omezenou množinou funkcí. Spustíte-li bash v režimu POSIX kompatibility, bude emulovat chování právě tohoto shellu.

bash neboli Bourne Again shell: standardní GNU shell, intuitivní a pružný. Asi nejvhod-nější pro začátečníky, současně však i mocný nástroj pro zkušené uživatele a profesioná-ly. V Linuxu je to standardní shell normálních uživatelů. Tento shell je zároveň takzvanou nadmnožinou Bourne shellu, doplňuje jej o různé přídavky a doplňky. Znamená to, že Bourne Again shell je kompatibilní s Bourne shellem. Cokoliv, co funguje v sh, bude fun-govat i v bashi. Opačně to samozřejmě neplatí. Všechny příklady a cvičení v této příruč-ce používají bash.
csh neboli C shell: svou syntaxí tento shell připomíná programovací jazyk C. Mají jej v obli-bě někteří programátoři.
tcsh neboli Turbo C shell: nadmnožina klasického C shellu, rychlejší a uživatelsky příjem-nější.
ksh neboli Korn shell: oceňovaný zejména lidmi s unixovou minulostí. Nadmnožina Bour-ne shellu, ve standardní konfiguraci noční můra začínajícího uživatele.
zsh je další podstatně vylepšený shell. Zajímavé informace v češtině najdete například v článku *http://www.root.cz/clanky/zuzo-shell/*.

Přehled shellů, které systém zná, naleznete v souboru /etc/shells:

```
mia:~> cat /etc/shells /bin/bash /bin/sh /bin/tcsh /bin/csh
```

Falešný Bourne shell

Soubor /bin/sh je ve většině systémů pouze odkaz na /bin/bash. Při spuštění prostřednic-tvím tohoto odkazu se bash přepíná do režimu kompatibility s Bourne shellem.

Výchozí shell každého uživatele je nastaven v souboru /etc/passwd; takto může vypadat přísluš-ný záznam uživatele *mia*:

```
mia:L2NOfqdlPrHwE:504:504:Mia Maya:/home/mia:/bin/bash
```

Pokud chcete přejít do jiného shellu, jednoduše v aktivním terminálu zadejte název nového shel-lu. Díky nastavení proměnné PATH systém nalezne adresář, ve kterém se požadovaný shell nachá-zí, a protože je shell uložen na disku jako běžný spustitelný soubor, dojde k jeho spuštění. Prav-děpodobně se objeví jiná výzva, protože každý shell má svůj typický vzhled:

mia:~> tcsh [*mia@post21 ~]$*

Který shell používám?

Pokud nevíte, který shell právě používáte, vyhledejte si příslušný řádek v souboru /etc/passwd anebo spusťte následující příkaz:

```
echo $SHELL
```

Domovský adresář

Domovský adresář je místo, ve kterém se uživatel standardně ocitne po přihlášení do systému. Ve většině případů jde o podadresář adresáře /home, nemusí to ale platit vždy. Domovský adresář může být umístěn na disku vzdáleného serveru, pak jej obvykle najdete jako /nethome/uživatel-ské_jméno. Správce systému může zvolit i komplikovanější rozvržení systému a svůj domovský adresář pak můžete najít například jako /disk6/HU/07/jgillard.

Ať už je cesta k vašemu domovskému adresáři jakákoliv, nemusíte se o ni příliš starat. Pokud by ji nějaký program potřeboval, je zapsána v proměnné prostředí HOME. Příkazem echo můžete obsah této proměnné zobrazit:

orlando:~> echo $HOME /nethome/orlando

Ve svém domovském adresáři si můžete dělat, co chcete. Můžete v něm vytvářet libovolné sou-bory a adresáře, i když jsou samozřejmě počty souborů a celková velikost uložených dat omeze-ny hardwarovými limity, velikostí diskového oddílu a někdy i kvótami, které nastavil administrá-tor systému. Nastavování limitů byla běžná praxe v dobách, kdy byl diskový prostor ještě drahý. Zda máte pro svůj adresář nějaké limity nastaveny, můžete zjistit příkazem quota:

pierre@lamaison:/> quota -v Diskquotas for user pierre (uid 501): none

Pokud jsou diskové kvóty nastaveny, vypíše se seznam oddílů s nastavenými omezeními a kon-krétně nastavená omezení. Kvóty mohou být nastaveny tak, že může být tolerováno krátkodobé překročení limitu. Podrobnější informace naleznete pomocí příkazů info quota nebo man quota.

Chybí příkaz quota

Pokud v systému neexistuje příkaz quota, pak diskové limity nastaveny nejsou.

Snadné nalezení domovského adresáře je možné pomocí vlnovky (~), která slouží jako zkratka cesty /cesta_k_domovskému_adresáři/uživatel. Je to stejná hodnota jako ta, která je uložena v pro-měnné HOME, o její nastavení se nemusíte nijak starat. Jednoduchý příklad: Z adresáře /var/music/albums/arno/2001 můžete přejít do adresáře s obrázky ve vašem domovském adresáři jediným elegantním příkazem:

rom:/var/music/albums/arno/2001> cd ~/images rom:~/images> pwd /home/rom/images

V další části této kapitoly budeme hovořit o příkazech pro manipulaci se soubory a adresáři, s jejichž pomocí můžete ve svém domovském adresáři udržet pořádek.

Nejdůležitější konfigurační soubory

Jak už bylo řečeno, většina konfiguračních souborů se ukládá v adresáři /etc. Obsah souborů je možno prohlížet příkazem cat, který vypíše textový soubor na standardní výstup (tedy obvykle na monitor). Jeho použití je velmi snadné:

cat file1 file2 ... fileN

V této části vám nabídneme přehled nejobvyklejších konfiguračních souborů. V žádném případě nepůjde o úplný seznam. Při instalaci nových programů mohou v adresáři /etc vzniknout nové konfigurační soubory. Až budete konfigurační soubory číst, zjistíte, že jsou většinou velmi dobře okomentované a samovysvětlující. Pro některé soubory dokonce existují vlastní manuálové strán-ky, které obsah souboru popisují, například man group.

Soubor/adresář	Informace/služba
aliases	Poštovní aliasy používané programy Sendmail nebo Postfix. Běžnou a dávnou unixovou tradicí je, že na téměř každém systému běží poštovní server, prakticky každá linuxová distribuce obsa-huje poštovní server Sendmail. V tomto souboru se lokální uživatelská jména mapují na e-mai-lové adresy, případně jiná lokální jména.
apache	Adresář s konfiguračními soubory webového serveru Apache.
bashrc	Konfigurace shellu Bash platná pro celý systém. Definuje funkce a aliasy platné pro všechny uživatele. Podobné konfigurace mívají i jiné shelly, například soubor cshrc.
crontab a adresáře cron.*	Konfigurace úloh, které se mají automaticky spouštět v pravidelných intervalech – zálohování, aktualizace systémových databází, čištění systému, rotace logů a podobně.
default	Výchozí chování některých příkazů, například .
filesystems	Známé souborové systémy: ext3, vfat, iso9660 a podobně.
fstab	Seznam diskových oddílů a jejich přípojných bodů.
ftp*	Konfigurace FTP serveru: Kdo se může připojit, které části systému jsou přístupné a podobně.
group	Konfigurace skupin uživatelů. K editaci tohoto souboru používejte příkazy , a . Do přímé editace se pouštějte pouze v případě, že přesně víte, co děláte.
hosts	Seznam počítačů dostupných v síti bez nutnosti použít službu DNS – jinak řečeno, mapování názvů vybraných počítačů na jejich IP adresy. Tento soubor nijak nesouvisí s konfigurací síového subsystému, který se nastavuje v adresáři /etc/sysconfig.
inittab	Konfigurace bootovacího procesu: režim, počet textových konzol a podobně.
issue	Informace o distribuci (verze, informace o jádře).

ld.so.conf	Umístění knihovních souborů.
lilo.conf, silo.conf, aboot.conf atd.	Nastavení zavaděče LInux LOader, programu, který zajišuje zavedení systému a v současnosti se obvykle nahrazuje zavaděčem GRUB.

Soubor/adresář	Informace/služba
logrotate.*	Nastavení rotace logů, mechanismu, který zabraňuje růstu velikosti logovacích souborů nade všechny meze.
mail	Adresář s nastaveními poštovního serveru.
modules.conf	Konfigurace modulů, které zajišují různé speciální funkce (ovladače).
motd	Zpráva dne. Zobrazuje se každému, kdo se přihlásí v textovém režimu. Administrátor může pomocí tohoto souboru oznamovat například plánovanou údržbu.
mtab	Přehled aktuálně připojených souborových systémů. Tento soubor se nedoporučuje editovat.
nsswitch.conf	Nastavuje pořadí použití jednotlivých služeb pro překlad názvů na adresy.
pam.d	Konfigurace autentizačních modulů.
passwd	Seznam lokálních uživatelů. K editaci tohoto souboru používejte příkazy , a . Do přímé editace se pouštějte pouze v případě, že přesně víte, co děláte.
printcap	Zastaralý, stále však často používaný soubor s konfigurací tiskáren. Pokud přesně nevíte, co děláte, nezkoušejte soubor editovat ručně.
profile	Celosystémově platná konfigurace prostředí shellu: proměnné, výchozí parametry souborů, limity využití prostředků a podobně.
rc*	Adresář s definicemi služeb pro jednotlivé úrovně běhu.
resolv.conf	Nastavení překladu jmen službou DNS.
sendmail.cf	Konfigurace poštovního serveru Sendmail.
services	Seznam síových služeb, které systém zná.
sndconfig nebo sound	Konfigurace zvukové karty a zvukových událostí.
ssh	Adresář s konfiguracemi klienta a serveru služby SSH.
sysconfig	Adresář se systémovými konfiguračními soubory, zde se nastavuje myš, klávesnice, sí, správa napájení a podobně. (Platí pro distribuci Red Hat.)
X11	Adresář s konfigurací grafického serveru X. Obsah adresáře závisí na používaném serveru, hlavní konfigurační soubor je bu xorg.conf nebo XFree86Config. Obsahuje rovněž nastavení instalovaných správců oken, například , , a podobně.
xinetd.* nebo inetd.conf	Konfigurační soubor internetových služeb, jejichž činnost zajišuje démon xinetd nebo inetd (tedy služeb, které neběží jako samostatné servery).

Nejobvyklejší konfigurační soubory

Postupně se o jednotlivých souborech budeme dozvídat více a budeme se podrobněji zabývat jejich obsahem.

Nejobvyklejší zařízení

Zařízení, obecně jakékoliv periferie připojené k počítači, se v systému reprezentují jako položka v adresáři /dev. Jednou z výhod tohoto unixového mechanismu pro manipulaci se zařízeními je, že se ani uživatel, ani systém nemusí příliš starat o přesné vlastnosti zařízení.

Uživatelé, kteří s Linuxem či Unixem přijdou do styku poprvé, bývají často zaskočeni velkým počtem nových názvů a celých konceptů, s nimiž se musí seznámit. Proto v této úvodní části uvádíme seznam obvyklých zařízení.

Název	Zařízení
cdrom	CD mechanika
console	Speciální záznam pro aktuálně používanou konzolu
cua*	Sériové porty (staré označení, novější je ttyS*)
dsp*	Zařízení pro vzorkování a záznam (zvuková karta)
fd*	Záznamy pro všechny možné druhy disketových mechanik, výchozí je /dev/fd0, odpovídající mechanice pro 1,44MB diskety.
hd[a-t][1-16]	Pevné disky IDE s maximálním možným počtem oddílů.
ir*	Infračervená zařízení
isdn*	ISDN spojení
js*	Joysticky
lp*	Tiskárny
mem	Pamě
midi*	Přehrávač MIDI
mixer* a music	Idealizovaný model mixéru (kombinuje zvukové signály)
modem	Modem
mouse (též msmouse, logimouse, psmouse, input/mice, psaux)	Různé typy myší
null	Bezedná černá díra
par*	Paralelní porty
pty*	Pseudo terminály
radio*	Radiová zařízení (HAM)
ram*	Bootovací zařízení
sd*	SCSI, příp. SATA, disky a jejich oddíly
sequencer	Syntezátor na zvukové kartě
tty*	Virtuální konzoly emulující terminál VT100
usb*	USB karty a skenery
video*	Grafické karty s podporou videa

Běžná zařízení

Nejobvyklejší proměnlivé soubory

V adresáři /var se nachází řada adresářů pro uložení často se měnících dat (ve smyslu srovnání s binárními programy nebo konfiguračními soubory, jejichž obsah se mění jen zřídka nebo vůbec). V adresáři /var se nacházejí soubory, jejichž obsah se mění velmi často, jako jsou například logo-vací záznamy, poštovní schránky, soubory zámků, tiskové fronty a podobně.

Z bezpečnostních důvodů se tyto soubory obvykle udržují odděleně od hlavního souborového systému, takže je můžeme snáze sledovat, případně k nim přísněji omezit přístup. Řada těchto sou-borů navíc naopak potřebuje mnohem volnější přístupová práva, například do adresáře /var/tmp musí mít právo zápisu každý. V těchto místech může panovat čilá uživatelská aktivita, mnohdy vyvolaná i zcela anonymními uživateli připojenými přes Internet. Právě proto se adresář /var včet-ně všech svých podadresářů obvykle vytváří na samostatném diskovém oddílu. Tímto způsobem se eliminují různá rizika, například že poštovní bomba zaplní diskový oddíl s důležitými daty, jako jsou programy nebo konfigurační soubory.

/var/tmp a /tmp

Soubory v adresáři /tmp mohou být kdykoliv smazány, ať už v rámci pravidelné údržby nebo při restartu systému. Na specificky nastaveném systému se takto může chovat i adre-sář /var/tmp, nicméně ve standardním nastavení k jeho nečekanému mazání nedochází. Proto jej lze použít k ukládání dočasných souborů. Pokud si nejste chováním systému jisti, zeptejte se příslušného administrátora. Jestliže systém spravujete sami, pak máte slušnou jistotu, že je adresář /var/tmp bezpečné místo – pokud jste ovšem (jako *root*) jeho chová-ní nějak specificky nezměnili.

Při jakýchkoliv činnostech se vždy snažte držet privilegií, která mají normální uživatelé – neukládejte soubory přímo do

kořene souborového systému ani do adresáře /usr či jiné-ho podadresáře v některé rezervované oblasti. Vyhnete se tak neúmyslnému poškození důležitých částí souborového systému.

Jedním z hlavních bezpečnostních mechanismů unixových systémů, který je přirozeně implemen-tován také v každém linuxovém systému, je logovací mechanismus, který zaznamenává aktivity uživatelů, procesů, systémové události a podobně. V konfiguračním souboru logovacího démona *syslog* se nastavuje, jaké události se kam zaznamenávají. Výchozím adresářem pro všechny tyto záznamy je /var/log, který obsahuje různé soubory se záznamy o přihlášeních, činnostech serve-rů a tak dále.

V adresáři /var se typicky nacházejí datové soubory různých serverů, které jsou tak uloženy oddě-leně od kritických dat, jako jsou samotné binární programy serverů a jejich konfigurační soubory. Typickým příkladem v linuxovém systému je adresář /var/www, který obsahuje HTML stránky, skripty a obrázky, poskytované webovým serverem. Stejně tak je rozumné do některého poda-dresáře ve /var umístit strom FTP serveru (tedy data, k nimž mají přístup vzdálení uživatelé). Pro-tože jde o veřejně přístupná data, která mohou měnit anonymní uživatelé, je bezpečnější umístit je zde, mimo oddíly a adresáře s citlivými daty.

Na většině pracovních stanic obsahuje adresář /var/spool přinejmenším podadresáře démonů at a cron, které obsahují naplánované úlohy. V kancelářském prostředí se zde obvykle nachází i adresář lpd, který obsahuje tiskovou frontu či fronty, konfigurační soubory tiskáren a záznamy
o tisku. Na serverech typicky nalezneme adresář /var/spool/mail, který obsahuje příchozí poštu lokálních uživatelů, roztříděnou do samostatného souboru pro každého uživatele. S poštovními službami souvisí i adresář mqueue, fronta neodeslaných správ. Na serveru obsluhujícím mnoho uživatelů se
v této oblasti může odehrávat velké množství aktivity. Ve větvi /var/spool se nachází i úložná oblast news serveru, který rovněž musí zpracovávat obrovské množství dat. Pro distribuce založené na balíčkovacím systému RPM (RedHat Package Manager) je charakteris
tický adresář /var/lib/rpm, kde se ukládají informace o balíčcích.

Manipulace se soubory

Zobrazení vlastností souboru Více o příkazu ls

Kromě samotného názvu souboru vypisuje příkaz ls i celou řadu dalších informací, jako je napří-klad typ souboru, o kterém jsme už hovořili. Může zobrazit rovněž přístupová práva souboru, veli-kost souboru, číslo inode, datum a čas vytvoření, vlastníka nebo počet odkazů vedoucích na sou-bor. Použijete-li příkaz ls s přepínačem -a, vypíší se i soubory, které jsou normálně před uživate-lem skryté. Jsou to soubory, jejichž název začíná tečkou – příkladem může být celá řada konfigu-račních souborů ve vašem domovském adresáři. Po nějaké době používání systému zjistíte, že vznikly desítky souborů a adresářů, které se ve výpisu obsahu adresáře automaticky neobjevují. Každý adresář navíc obsahuje soubory pojmenované . a .., z jejichž čísla inode systém dokáže odvodit umístění adresáře v hierarchii souborového stromu.

Rozhodně byste si měli přečíst informační stránku příkazu ls, protože jde o velmi běžný příkaz s celou řadou užitečných přepínačů. Přepínače je možno vzájemně kombinovat, jak je to u větši-ny unixových příkazů obvyklé. Běžná je například kombinace ls -al, která vypíše dlouhý seznam souborů a jejich vlastností včetně cílů, na něž míří symbolické odkazy. Příkaz ls -latr vypíše stej-ný seznam, ovšem uspořádaný podle data poslední změny, takže naposledy změněné soubory budou uvedeny na konci výpisu. Ukažme si několik příkladů:

```
krissie:~/mp3> lsAlbums/ Radio/ Singles/ gene/ index.html

krissie:~/mp3> ls -a./ .thumbs Radio gene/../ Albums/ Singles/ index.html

krissie:~/mp3> ls -l Radio/total 8drwxr-xr-x 2 krissie krissie 4096 Oct 30 1999 Carolina/drwxr-xr-x 2 krissie krissie 4096 Sep 24 1999 Slashdot/

krissie:~/mp3> ls -ld Radio/drwxr-xr-x 4 krissie krissie 4096 Oct 30 1999 Radio/

krissie:~/mp3> ls -ltrtotal 20drwxr-xr-x 4 krissie krissie 4096 Oct 30 1999 Radio/-rw-r--r--1 krissie krissie 453 Jan 7
2001 index.htmldrwxrwxr-x 30 krissie krissie 4096 Oct 20 17:32 Singles/drwxr-xr-x 2 krissie krissie 4096 Dec 4
23:22 gene/drwxrwxr-x 13 krissie krissie 4096 Dec 21 11:40 Albums/
```

Na většině linuxových systémů je příkaz ls *aliasem* na barevnou verzi příkazu ls. Díky tomu jsou různé typy souborů názorně vidět bez nutnosti používat další přepínače, protože každý typ je vypsán odlišnou barvou. Standardní barevné schéma je definováno v souboru /etc/DIR_COLORS: Podrobnější informace naleznete v manuálových stránkách. Dříve se stejné informace znázorňo-valy pomocí speciálních znaků, doplňovaných za název souboru. Při jednobarevném zobrazení (například při tisku výpisu na tiskárnu) a obecně kvůli lepší čitelnosti se tento mechanismus pou-žívá dodnes:

Barva	Typ souboru

modrá	adresáře
červená	komprimované archivy
bílá	textové soubory
růžová	obrázky
azurová	odkazy
žlutá	zařízení
zelená	spustitelné soubory
blikající červená	neplatné odkazy

Výchozí barevné schéma příkazu ls

Znak	Typ souboru
žádný	normální soubor
/	adresář
*	spustitelný soubor
@	odkaz
=	soket
|	roura

Indikace typu souboru přídavnými znaky

Popis všech funkcí a možností příkazu ls můžete získat příkazem info ls.

Další nástroje

Více informací o tom, jaký typ dat soubor obsahuje, můžete zjistit příkazem file. Pomocí různých testů, které vyhodnotí vlastnosti souboru, magická čísla (viz man file) a jazykové charakteristiky, je příkaz file schopen kvalifikovaně odhadnout formát souboru. Ukažme si několik příkladů:

```
mike:~> file Documents/ Documents/: directory

mike:~> file high-tech-stats.pdf high-tech-stats.pdf: PDF document, version 1.2

mike:~> file Nari-288.rm Nari-288.rm: RealMedia file

mike:~> file bijlage10.sdw bijlage10.sdw: Microsoft Office Document

mike:~> file logo.xcf logo.xcf: GIMP XCF image data, version 0, 150 x 38, RGB Color mike:~> file cv.txt cv.txt: ISO-8859 text

mike:~> file image.png image.png: PNG image data, 616 x 862, 8-bit grayscale, non-interlaced

mike:~> file figure figure: ASCII text

    mike:~> file me+tux.jpg me+tux.jpg: JPEG image data, JFIF standard 1.01,
    resolution (DPI), "28 Jun 1999", 144 x 144

  mike:~> file 42.zip.gz 42.zip.gz: gzip compressed data, deflated, original filename,
  `42.zip', last modified: Thu Nov 1 23:45:39 2001, os: Unix

mike:~> file vi.gif vi.gif: GIF image data, version 89a, 88 x 31

mike:~> file slide1 slide1: HTML document text

mike:~> file template.xls template.xls: Microsoft Office Document

mike:~> file abook.ps abook.ps: PostScript document text conforming at level 2.0

mike:~> file /dev/log /dev/log: socket

mike:~> file /dev/hda /dev/hda: block special (3/0)
```

Příkaz file používá řadu přepínačů, například -z, kterým se zapne režim prohlížení komprimova-ných souborů. Více podrobností viz info file. Nezapomínejte, že údaj příkazu file není stopro-centní, jde pouze o odhad. Jinak řečeno, není nemožné příkaz file přelstít.

Proč ten povyk kolem typů a formátů souborů

Zanedlouho budeme hovořit o několika nástrojích pro práci s *prostými textovými soubory*. Pokud tyto nástroje použijete na jiný typ souboru, nebudou fungovat. V nejhorším přípa-dě můžou způsobit havárii terminálu a vygenerovat spoustu pípání. Pokud se vám to stane, prostě zavřete okno terminálu a otevřete nové. Snažte se ale takovým situacím vyhnout.

Vytváření a mazání souborů a adresářů Vyrobit chaos...

...není nijak zvlášť složité. Dnes je většina počítačů zapojena do sítě a přirozeně se tak kopírují soubory z jednoho počítače na druhý. Zejména při práci v grafickém prostředí je vytvoření nové-ho souboru úplně triviální a často se tak děje bez vědomí uživatele. Abychom demonstrovali, o čem hovoříme, takto vypadá obsah adresáře nově vytvořeného uživatele na standardním systému RedHat:

[newuser@blob user]$ ls -al total 32 drwx------3 user user 4096 Jan 16 13:32 . drwxr-xr-x 6 root root 4096 Jan 16 13:32 .. -rw-r--r--1 user user 24 Jan 16 13:32 .bash_logout -rw-r--r--1 user user 191 Jan 16 13:32 .bash_profile -rw-r--r--1 user user 124 Jan 16 13:32 .bashrc drwxr-xr-x 3 user user 4096 Jan 16 13:32 .kde -rw-r--r--1 user user 3511 Jan 16 13:32 .screenrc -rw-------1 user user 61 Jan 16 13:32 .xauthDqztLr

Výpis obsahu „použitého“ adresáře nevypadá na první pohled o moc hůř:

olduser:~> lsapp-defaults/ crossover/ Fvwm@ mp3/ OpenOffice.org638/articles/ Desktop/ GNUstep/ Nautilus/ staroffice6.0/bin/ Desktop1/ images/ nqc/ training/brol/ desktoptest/ Machines@ ns_imap/ webstart/C/ Documents/ mail/ nsmail/ xml/closed/ Emacs@ Mail/ office52/ Xrootenv.0

Pokud bychom ovšem vypsali i adresáře a soubory začínající tečkou, zjistíme, že adresář obsahu-je 185 položek. Je to dáno tím, že většina aplikací si v domovském adresáři každého uživatele vytváří vlastní adresáře a/nebo soubory s nastaveními daného uživatele. Obvykle se tyto soubory vytvoří při prvním spuštění programu. Občas vás některé programy upozorní, že adresář s nasta-veními neexistuje, ve většině případů ale dojde k jeho vytvoření bez jakéhokoliv upozornění.

Plynule navíc vznikají další a další soubory, protože uživatelé chtějí ukládat své dokumenty, archi-vovat jejich různé verze, používat internetové aplikace a stahovat na počítač soubory a přílohy. Nikdy to nekončí. Je proto jasné, že potřebujeme nějaké mechanismy, jak si v tom udržet přehled.

V následující části popíšeme naši představu udržování pořádku. Budeme hovořit pouze o texto-vých nástrojích dostupných v shellu, protože existující grafické nástroje jsou velmi intuitivní a vypadají a chovají se stejně jako známé klikací souborové manažery z MS Windows, včetně gra-fické nápovědy a dalších ozdob, které se od takového typu aplikace očekávají. Následující seznam představuje přehled nejoblíbenějších souborových manažerů v GNU/Linuxu. Většinu z nich spus-títe buď z nabídky na pracovní ploše nebo klepnutím na ikonu domovského adresáře, případně následujícím příkazem zadaným na příkazovém řádku:

nautilus: výchozí souborový manažer používaný v Gnome. Vynikající dokumentaci k tomuto nástroji naleznete na adrese *http://www.gnome.org.*
konqueror: souborový manažer používaný v prostředí KDE. Příručku najdete na adrese

http://docs.kde.org.

- mc: Midnight Commander, souborový manažer inspirovaný Norton Commanderem. Doku-mentaci naleznete na adrese *http://gnu.org/directory* nebo na některém ze zrcadel, napří-klad *http://www.ibiblio.org.*

Tyto nástroje rozhodně stojí za vyzkoušení a obvykle na každého začínajícího uživatele Linuxu udělají velký dojem, kdyby pro nic jiného, tak pro svou rozmanitost. Jde pouze o tři nejznámější souborové manažery, kromě nich existuje i celá řada dalších podobných projektů. Podívejme se nyní podrobněji na operace se souborovým systémem a ukažme si, které základní unixové příka-zy jsou v těchto grafických nástrojích „schovány“.

Nástroje

Vytváření adresářů

Jedna z možností, jak udržet věci tam, kde patří, je přidělit různým souborům různá standardní místa uložení prostřednictvím adresářů a podadresářů (nebo složek a podsložek, chcete-li). Adre-sáře se vytvářejí příkazem mkdir:

richard:~> mkdir archive

richard:~> ls -ld archivedrwxrwxrwx 2 richard richard 4096 Jan 13 14:09 archive/

Adresář i podadresář lze vytvořit v jednom kroku pomocí přepínače -p:

```
richard:~> cd archive

richard:~/archive> mkdir 1999 2000 2001

richard:~/archive> ls 1999/ 2000/ 2001/

richard:~/archive> mkdir 2001/reports/Restaurants-Michelin/ mkdir: cannot create directory `2001/reports/Restaurants-
Michelin/': No such file or directory

richard:~/archive> mkdir -p 2001/reports/Restaurants-Michelin/

richard:~/archive> ls 2001/reports/ Restaurants-Michelin/
```

Pokud potřebujete vytvořit adresář s jinými oprávněními, než jsou výchozí nastavená, můžete to stále udělat jediným příkazem mkdir, podrobnosti viz informační stránky. O přístupových prá-vech budeme hovořit v následující části, věnované zabezpečení souborů.

Pro názvy adresářů platí stejná pravidla jako pro názvy souborů. Nejdůležitějším omezením je, že v rámci jednoho adresáře nemůžete mít vytvořeny dva soubory se stejným názvem (nezapomeň-te ale, že Linux, stejně jako UNIX, rozlišuje v názvech souborů mezi velkými a malými písmeny). Délka názvu není omezena, doporučuje se ale nepřekročit 80 znaků, aby se název souboru vešel na jeden řádek obrazovky. V názvech souborů můžete používat libovolné znaky, doporučuje se ale nepoužívat znaky, které mají v shellu speciální význam. V případě pochybností se podívejte do přílohy C.

Přesouvání souborů

Nyní máme v domovském adresáři vytvořenu přehlednou strukturu podadresářů, takže je čas roz-místit do nich nezařazené soubory pomocí příkazu mv:

```
richard:~/archive> mv ../report[1-4].doc reports/Restaurants-Michelin/
```

Stejným příkazem je možné soubory přejmenovat:

```
richard:~> ls To_Do -rw-rw-r--1 richard richard 2534 Jan 15 12:39 To_Do

richard:~> mv To_Do done

richard:~> ls -l done -rw-rw-r--1 richard richard 2534 Jan 15 12:39 done
```

Je zřejmé, že se změnil pouze název souboru, všechny ostatní vlastnosti zůstaly beze změny. Podrobné informace o syntaxi a možnostech příkazu mv naleznete v manuálových a informač-ních stránkách. Tuto dokumentaci byste měli instinktivně použít pokaždé, když narazíte na něja-ký problém. Řešení problému je v dokumentaci s největší pravděpodobností popsáno. I zkušený uživatel používá manuálové stránky denně, začínající uživatel by je měl proto používat pořád. Zanedlouho si zapamatujete nejčastější volby běžně používaných příkazů, dokumentace však bude i nadále představovat primární zdroj informací. Informace uvedené na manuálových strán-kách, v dokumentech HOWTO a FAQ jsou postupně převáděny do informačních stránek, které dnes představují nejaktuálnější (a trvale dostupný) zdroj informací.

Kopírování souborů

Soubory a adresáře se kopírují příkazem cp. Užitečným přepínačem je -R, který zapíná rekurziv-ní kopírování (tedy kopírování všech podřízených souborů a podadresářů). Obecná syntaxe pří-kazu cp je následující:

```
cp [-R] zdrojový_soubor cílový_soubor
```

Ukažme si příklad uživatele *newguy*, který chce mít stejná nastavení grafického prostředí Gnome, jako má uživatel *oldguy*. Jedna z možností je zkopírovat všechna nastavení z domovského adre-sáře uživatele *oldguy* do domovského adresáře uživatele *newguy*:

```
victor:~> cp -R ../oldguy/.gnome/ .
```

Takto spuštěný příkaz vypíše několik chybových hlášení o přístupových právech, ta se však ves-měs týkají privátních souborů, které uživatel *newguy* tak jako tak nepotřebuje. Pokud by zkopí-rovaná nastavení i přesto nefungovala, o změně přístupových práv budeme hovořit v následující části.

Mazání souborů

Příkazem rm smažete jeden soubor, příkazem rmdir zase jeden prázdný adresář. (Příkazem lsa můžete ověřit, zda adresář je či není prázdný.) Příkaz rm nabízí i přepínače pro mazání neprázdných adresářů včetně jejich podadresářů, podrobnosti o této značně nebezpečné volbě naleznete v informačních stránkách.

Jak prázdný může být adresář

Je jasné, že z adresáře nelze odstranit položky . a .., protože ty definují umístění (i prázd
ného) adresáře v hierarchii adresářového stromu.

V Linuxu, stejně jako v Unixu, neexistuje odpadkový koš – přinejmenším ne přímo v shellu, pro grafická prostředí existuje celá řada jeho implementací. Jakmile soubor jednou smažete, je prostě pryč a neexistuje žádný obecný způsob, jak jej zachránit – pomohou vám jenom zálohy, anebo, s velkou dávkou štěstí, pokud jste hodně rychlí a máte hodně dobrého správce systému. Jako ochrana před takovým omylem slouží u příkazů rm, cp a mv přepínač -i. Pokud jej použijete, příkaz neprovede požadovanou operaci okamžitě, ale vyžádá si potvrzení. Od katastrofy vás tak dělí jeden stisk klávesy Enter navíc:

mary:~> rm -ri archive/ rm: descend into directory `archive'? y rm: descend into directory `archive/reports'? y rm: remove directory `archive/reports'? y rm: descend into directory `archive/backup'? y rm: remove `archive/backup/sysbup200112.tar'? y rm: remove directory `archive/backup'? y rm: remove directory `archive'? y

Tento přepínač lze nastavit i jako implicitně zapnutý. Budeme o tom hovořit v kapitole „Home, sweet /home“, kde popisujeme vlastní nastavení prostředí shellu.

Hledání souborů Využití funkcí shellu

V příkladu na přesouvání souborů už jsme si ukázali, jak shell dokáže pracovat s více souborynajednou. V uvedeném příkladu shell automaticky zjistil, co uživatel myslí požadavkem mezi hra-natými závorkami „[“ a „]“. Stejným způsobem může shell substituovat jak rozsah číslic, tak vel-kých či malých písmen. Hvězdička pak slouží jako náhrada libovolného počtu znaků, otazníknahrazuje právě jeden znak.

Jednotlivé způsoby substituce lze vzájemně kombinovat, shell je v tomto směru velmi logický.

Bash například nebude mít problém se zápisem ls adresář/*/*/*[2-3].V jiných shellech se hvězdička běžně používá k úspoře při psaní – stačí například napsat cd adr*namísto cd adresář. V bashi to ovšem není nutné, protože GNU shell nabízí funkci automatické-ho dokončování. Stačí tak napsat jen několik prvních znaků příkazu (umístěného kdekoliv) nebosouboru (umístěného v aktuálním adresáři), a pokud je tím název určen jednoznačně, shell zjistí,co jste měli na mysli. Pokud například adresář obsahuje mnoho souborů, můžete snadno zjistit,zda obsahuje nějaké soubory začínající znakem A tak, že napíšete ls A a dvakrát zmáčknete klá-vesu Tab. Pokud je v adresáři jediný soubor začínající znakem A, bude jeho název ihned doplněnjako parametr příkazu ls (nebo libovolného jiného příkazu).

Which

Jednoduchý nástroj pro hledání souborů představuje příkaz which, který prohledává adresáře uvedené v uživatelem nastavené cestě. Samozřejmě tato cesta obsahuje pouze seznam adresářů se spustitelnými soubory, takže příkaz which nelze použít ke hledání běžných souborů. Příkaz which je užitečným nástrojem při hledání příčiny chybového hlášení „Command not found“. V následujícím příkladu nemůže uživatelka *tina* spustit příkaz acroread, zatímco její kolega nemá se spuštěním příkazu problém. Je to podobná situace jako příklad v části věnované proměnné PATH. Kolega Tině řekl, že program spouští z adresáře /opt/acroread/bin (zjistil to příkazem which acroread), ona však tento adresář v cestě nastaven nemá:

tina:~> which acroread/usr/bin/which: no acroread in (/bin:/usr/bin:/usr/bin/X11)

Tina může problém vyřešit tak, že bude program acroread spouštět s uvedením celé cesty nebo tím, že si upraví obsah proměnné PATH:

tina:~> export PATH=$PATH:/opt/acroread/bin

tina:~> echo $PATH /bin:/usr/bin:/usr/bin/X11:/opt/acroread/bin

Pomocí příkazu which je také možné snadno zjistit, zda nějaký příkaz není aliasem jiného:

gerrit:~> which -a ls ls is aliased to `ls -F --color=auto' ls is /bin/ls

Find a locate

Tyto dva příkazy představují univerzální nástroje pro hledání souborů mimo spouštěcí cestu. Pří-kaz find je známý už z Unixu a je velmi mocný, i když jeho syntaxe je poněkud komplikovaněj-ší. GNU verze příkazu find ale používá rozumnou syntaxi. Pomocí tohoto příkazu můžete hledat soubory nejen podle názvu, ale i podle velikosti, data poslední změny a jiných vlastností soubo-ru. Nejčastěji se používá k hledání podle názvu:

find <cesta> -name <řetězec>

Tímto zápisem příkazu nařizujeme prohledat všechny adresáře v a pod zadanou cestou a vypsat názvy těch souborů, v jejichž názvu se objevuje zadaný řetězec (prohledávají se a vypisují se názvy souborů, nikoliv jejich obsah).

Příkaz lze použít také ke hledání souborů o určité velikosti, jak to ukazuje následující příklad, kde uživatel v aktuálním adresáři a jeho podadresářích hledá soubory větší než 5 MB:

```
peter:~> find . -size +5000k psychotic_chaos.mp3
```

V manuálových stránkách také zjistíte, že prostřednictvím příkazu find můžete s nalezenými sou-bory provést zadanou operaci. Obvyklým příkladem je mazání souborů. Je rozumné nejprve pří-kaz spustit bez volby -exec, a teprve když si ověříte, že nalezl skutečně správné soubory, spustit jej znovu a specifikovat požadovanou akci. Následující příklad hledá a maže soubory, jejichž název končí na .tmp:

```
peter:~> find . -name "*.tmp" -exec rm {} \;
```

> Optimalizujte!
>
> Výše uvedený příklad spustí příkaz rm tolikrát, kolik bude nalezených souborů, což může být obecně velmi mnohokrát. To vede ke zbytečnému zatížení systému.
>
> Realističtější příklad by používal rouru (|) a příkaz xargs s příkazem rm jako parametrem. V takovém případě by se rm zavolal až po naplnění celého příkazového řádku, nikoliv samostatně pro každý soubor. O řešení běžných úloh prostřednictvím přesměrování vstup ně-výstupních operací budeme hovořit v kapitole „Přesměrování vstupu a výstupu“.

O nějaký čas později (podle manuálových stránek v roce 1999, tedy o 20 let později) než find vznikl příkaz locate. Tento příkaz se snáze používá, jeho možnosti jsou ale v porovnání s příka-zem find omezenější, protože pracuje s indexovanou databází souborů, která se aktualizuje jen jednou denně. Na druhé straně, prohledávání databáze je mnohem rychlejší než prohledávání sou-borového systému, zatížení počítače je nižší a uživatel vidí výsledek okamžitě.
Většina linuxových distribucí dnes používá příkaz slocate, což je modernější a bezpečnější verze příkazu locate, která před uživatelem skrývá soubory, jež nemá právo vidět. Příkladem takových souborů mohou být soubory v domovském adresáři superuživatele, které nejsou pro normální uži-vatele viditelné. Pokud bude chtít nějaký uživatel najít někoho, kdo něco ví o C shellu, může vyzkoušet příkaz locate .cshrc, jímž zjistí uživatele, kteří mají vlastní upravený konfigurační sou-bor C shellu. Pokud C shell používají uživatelé *root* a *jenny*, bude nalezen pouze soubor /home/jenny/.cshrc, soubor v domovském adresáři superuživatele viditelný nebude. Na většině systémů je příkaz locate pouze symbolický odkaz na program slocate:

```
billy:~> ls -l /usr/bin/locatelrwxrwxrwx 1 root slocate 7 Oct 28 14:18 /usr/bin/locate -> slocate*
```

Tina mohla pomocí programu locate najít program, který potřebovala:

```
tina:~> locate
acroread /usr/share/icons/hicolor/16x16/apps/acroread.png /usr/share/icons/hicolor/32x32/apps/acrore
ad.png /usr/share/icons/locolor/16x16/apps/acroread.png /usr/share/icons/locolor/32x32/apps/acroread
.png /usr/local/bin/acroread /usr/local/Acrobat4/Reader/intellinux/bin/acroread /usr/local/Acrobat4/bin
/acroread
```

Adresáře, v jejichž názvu není text bin, nemohou tento program obsahovat – neobsahují spusti-telné soubory. To, co Tina hledá, je soubor /usr/local/bin, je to odkaz na shellový skript, který spouští samotný program:

```
tina:~> file /usr/local/bin/acroread /usr/local/bin/acroread: symbolic link to ../Acrobat4/bin/acroread

tina:~> file /usr/local/Acrobat4/bin/acroread /usr/local/Acrobat4/bin/acroread: Bourne shell script text executable

tina:~> file /usr/local/Acrobat4/Reader/intellinux/bin/acroread /usr/local/Acrobat4/Reader/intellinux/bin/acroread: ELF 32-bit
LSB executable, Intel 80386, version 1, dynamically linked (uses shared libs), not stripped
```

Aby se udržela rozumná délka prohledávací cesty a systém nemusel prohledávat příliš mnoho adresářů pokaždé, když chce uživatel spustit program, přidává se do cesty jen adresář /usr/local/bin a nikoliv konkrétní adresáře obsahující jeden určitý spustitelný soubor. Adresář /usr/local/bin může obsahovat odkazy na tyto specifické programy a zároveň řadu dalších užitečných programů.

Jako obvykle můžete další podrobnosti o příkazech find a locate nalézt na manuálových a infor-mačních stránkách.

Příkaz grep

Obecné filtrování řádkového výstupu

Program grep je velmi jednoduchý, ale mocný program, který filtruje řádky vstupního textu a na výstup vypisuje pouze ty, které

obsahují zadaný řetězec. Tento program se používá na tisíc různých způsobů. Následující příklad ukazuje, jak uživatel hledá použití příkazu find v historii příkazů:

```
jerry:~> grep -a find .bash_history find . -name userinfo man find find ../ -name common.cfg
```

Prohlížení historie

V takovýchto situacích je užitečná i funkce bashe přímo určená k prohledávání historie pří-kazů, kterou aktivujete stiskem Ctrl+R. Takto by vypadalo její použití při hledání příkazu find:

```
thomas ~> ^R(reverse-i-search)`find': find `/home/thomas` -name *.xml
```

Po stisku Ctrl+R se objeví výzva k zadání hledaného textu. Prochází se historie příkazů aktuální relace (při ukončení relace se historie zapíše do souboru .bash_history) a zobra-zují se nejnovější výskyty hledaného řetězce. Pokud vás zajímají starší příkazy obsahující stejný řetězec, znovu zmáčkněte Ctrl+R.

Další podrobnosti najdete na informačních stránkách bashe.

Všechny trochu slušnější Unixy mají elektronický slovník. Stejně tak i Linux. Slovník je jednoduše seznam známých slov uložených v souboru words v adresáři /usr/share/dict. Chcete-li rychle ově-řit, jak se určité slovo píše, nepotřebujete k tomu žádný grafický nástroj:

```
william:~> grep pinguin /usr/share/dict/words

william:~> grep penguin /usr/share/dict/words penguin penguins
```

Jak se jmenuje ten chlapík, co má domovský adresář hned vedle? A hele, tady je jeho telefon!

```
lisa:~> grep gdbruyne /etc/passwd gdbruyne:x:981:981:Guy Debruyne, tel 203234:/home/gdbruyne:/bin/bash
```

A jakou že to má Arno e-mailovou adresu?

```
serge:~/mail> grep -i arno *sent-mail: To: <Arno.Hintjens@celeb.com>sent-mail: On Mon, 24 Dec 2001, Arno.Hintjens@celeb.com wrote:
```

Příkaz grep se velmi často používá v kombinaci s příkazy find a locate, což umožňuje vytvářet poměrně složité dotazy. Více informací naleznete v kapitole „Přesměrování vstupu a výstupu", věnované přesměrováním vstupně-výstupních operací.

Speciální znaky

Znaky, které mají pro shell nějaký speciální význam, je nutné při běžném použití *escapovat* – čili potlačit jejich původní význam. V bashi, stejně jako ve většině jiných shellů, k tomu slouží obrá-cené lomítko, které ruší speciální význam bezprostředně následujícího znaku. Shell rozlišuje celou řadu speciálních znaků, nejznámější jsou /, ., ? a *. Úplný seznam naleznete v informačních strán-kách a v dokumentaci k shellu.
Řekněme, že budete chtít zobrazit soubor „*", nikoliv všechny soubory v adresáři. Pak budete muset zadat:

```
less \*
```

Obdobně se pracuje se soubory, jejichž název obsahuje mezeru:

```
cat Tento\ Soubor
```

Další způsoby zobrazení obsahu souboru Obecně

Kromě příkazu cat, který v zásadě nedělá nic jiného, než že obsah souboru pošle na standardní výstup, existuje i řada dalších nástrojů pro prohlížení obsahu souborů.Nejjednodušší způsob je samozřejmě použít nějaký grafický program a nikoliv textový nástroj.V úvodní části už jsme viděli ukázku kancelářské aplikace, OpenOffice. Dalšími příklady mohoubýt: GIMP (spustíte jej z příkazového řádku zadáním příkazu gimp), The GNU Image Manipula-tion Program pro práci s rastrovými obrázky, xpdf k prohlížení dokumentů PDF, GhostView (gv)k prohlížení souborů PostScript, Mozilla/Firefox, links, Konqueror, Opera a řada dalších k pro-hlížení webových stránek, XMMS, CDplay a další pro práci s multimediálním obsahem, AbiWord,Gnumeric, KOffice a další pro kancelářské účely a celá řada jiných. Pro Linux existují tisíce apli-kací, vyjmenovat je by trvalo několik dnů.

My se ovšem budeme soustředit na textové aplikace, které představují základ pro všechny dalšíaplikace. Tyto příkazy nejlépe fungují v textovém režimu při prohlížení textových souborů. Jste-lina pochybách, zkontrolujte soubor nejprve příkazem file.

Podívejme se tedy, které textové nástroje můžeme použít ke zobrazení obsahu souborů.

Problémy s fonty

Nástroje, o nichž budeme dále hovořit, mohou mít problémy i s prostými textovými sou-bory kvůli kódování, které tyto soubory mohou používat. Speciální znaky, jako například znaky národních diakritik, čínské znaky a podobně, používají různé mechanismy kódová-ní a při pokusu o jejich zobrazení se velmi často místo nich vypíší nesmysly. O těchto problémech hovoříme v kapitole „Národní nastavení“.

less

Nejprve trocha historie.

Jako první vznikl příkaz cat. Ten vypíše celý obsah souboru najednou.
Pak se objevil příkaz pg, který dodnes najdete na starších Unixech. Tento příkaz vypisuje text po jednotlivých stránkách
Následoval program more, přepracovaná verze programu pg. Najdete jej na každém linu-xovém systému
Konečně program less je GNU verzí programu more a nabízí celou řadu funkcí jako zvý-razňovaní hledaných řetězců, posun v textu vpřed i vzad a podobně

Jeho syntaxe je velice jednoduchá:

less *název_souboru*

Více podrobností naleznete na informačních stránkách.
O těchto takzvaných stránkovacích programech jsme už hovořili v souvislosti se zobrazením manuálových stránek.

head a tail

Tyto dva příkazy zobrazí prvních, respektive posledních, *n* řádků souboru. Posledních deset zada-ných příkazů tak můžete zobrazit například takto:

```
tony:~> tail -10 .bash_history locate configure | grep bin man bash cd xawtv & grep usable /usr/share/dict/words grep
advisable /usr/share/dict/words info quota man quota echo $PATH frm
```

Příkaz head funguje obdobně. Příkaz tail navíc nabízí užitečnou funkci průběžného sledování obsahu, který do souboru přibývá. Toto chování zapnete přepínačem -f a velmi často je používa-jí administrátoři ke sledování systémových logů. Další informace najdete na informačních strán-kách.

Odkazy Typy odkazů

O souborech a reprezentaci souborů v souborovém systému toho už víme poměrně hodně, takže zvládnutí odkazů (či zástupců) bude maličkost. Odkaz není nic jiného než mechanismus umož-ňující, aby dva nebo více různých názvů souborů označovaly stejná fyzická data. Lze to provést dvěma metodami:

- Pevné odkazy (*hardlinks*): Asociují dva a více názvů souborů s jedním inodem. Pevné odkazy sdílejí stejnou datovou oblast disku, chovají se ale jako nezávislé soubory Z podstaty věci vyplývá jedna nevýhoda: Pevné odkazy nemohou mířit mimo svůj disko-vý oddíl, protože unikátnost inodů je zajištěna pouze v rámci jednoho oddílu.
- Symbolické odkazy (*softlinks, symlinks*): Krátký soubor, který je ukazatelem na jiný sou-bor. Symbolický odkaz obsahuje cestu k cílovému souboru, nikoliv odkaz na jeho umístě-ní na disku. Protože se v tomto mechanismu nepoužívají inody, mohou odkazy mířit i mimo svůj diskový oddíl

Oba druhy odkazů se chovají stejně, nejsou ale stejné, jak ukazuje obrázky na následující straně.

Jakmile odstraníte cílový soubor symbolického odkazu, je odkaz k ničemu.Každý normální soubor je ve své podstatě pevným odkazem. Pevné odkazy nemohou mířit mimosvůj diskový oddíl, protože se odkazují na čísla inodů a ta jsou jednoznačná vždy jen v rámci jed-noho diskového oddílu.

Lze namítnout, že existuje i třetí typ odkazů, takzvané *uživatelské odkazy*, jejichž princip je podobný zástupcům v MS Windows. Jde o soubory, které obsahují metadata interpretovatelná pouze grafickým souborovým manažerem. Z pohledu jádra a shellu jde o obyčejné datové sou-bory. Jejich název obvykle končí *.desktop* nebo *.lnk*, příklad můžete najít v adresáři ~/.gnome-deskop:

```
[dupont@boulot .gnome-desktop]$ cat La\ Maison\ Dupont [Desktop Entry] Encoding=Legacy-Mixed Name=La Maison Dupont Type=X-
nautilus-home X-Nautilus-Icon=temp-home URL=file:///home/dupont
```

A takto vypadá příklad z prostředí KDE:

```
[lena@venus Desktop]$ cat camera [Desktop Entry] Dev=/dev/sda1 FSType=auto Icon=memory MountPoint=/mnt/camera Type=FSDevice X-
KDE-Dynamic-Device=true
```

Vytváření těchto odkazů je v grafickém prostředí velmi jednoduché. Pokud byste potřebovalipomoc, určitě ji najdete v dokumentaci.V následující části se budeme věnovat vytváření symbolických odkazů v textovém prostředí.

Vytvoření symbolického odkazu

Symbolické odkazy jsou pro začínající uživatele velmi zajímavé – jsou snadno poznat a při jejichvytváření se není nutné zabývat diskovými oddíly.Odkazy se vytvářejí příkazem ln; chcete-li vytvořit symbolický odkaz, musíte uvést přepínač -s:
ln -s *cíl_odkazu název_odkazu*

V následujícím příkladu vytváří uživatel v podadresáři svého domovského adresáře odkaz na adre-sář v jiné části systému:

freddy:~/music> ln -s /opt/mp3/Queen/ Queen

freddy:~/music> ls -l lrwxrwxrwx 1 freddy freddy 17 Jan 22 11:07 Queen -> /opt/mp3/Queen

Symbolické odkazy jsou velmi malé soubory, velikost pevných odkazů odpovídá velikosti cílové
ho souboru. Symbolické odkazy jsou velmi užitečné. Často se používají kvůli úspoře místa, například pokud nově instalovaný program vyžaduje existenci určitého souboru na určitém místě, a vy jej máte umístěn jinde. Lze jimi „opravit“ skripty, které se najednou používají v jiném prostředí, a obecně mohou ušetřit spoustu práce. Správce systému může například přestěhovat domovské adresáře uživatelů na nové místo, třeba na disk2, a pokud chce, aby všechno (například soubor /etc/passwd) i nadále fungovalo správně, s minimální námahou může vytvořit symbolický odkaz /home mířící na /disk2/home.

Zabezpečení souborů

Přístupová práva: primární obranná linie

Bezpečnostní model Linuxu vychází z modelu používaného v unixových systémech a stejně jako tento velmi robustní model je i linuxová implementace velmi rigidní. V linuxovém systému je každý soubor vlastněn jedním uživatelem a jednou skupinou uživatelů. Ve vztahu k danému sou-boru pak existuje ještě třetí kategorie uživatelů, a to uživatelé, kteří nejsou vlastníkem a nejsou členy vlastnické skupiny. Pro každou z těchto tří kategorií uživatelů může mít soubor nastaveno nebo nenastaveno právo čtení, zápisu a spuštění.

Dlouhý výpis souborů příkazem ls -l už známe, i když nás prozatím zajímal z jiných důvodů. Tento příkaz zobrazuje rovněž přístupová práva všech tří uživatelských kategorií, jsou popsána devíti znaky, které bezprostředně následují za prvním znakem, udávajícím typ souboru. První tro-jice znaků popisuje práva vlastníka souboru, druhá trojice práva vlastnické skupiny a třetí trojice práva ostatních uživatelů. V rámci každé trojice jsou práva vypisována vždy ve stejném pořadí – čtení (*read*), zápis (*write*) a spuštění (*execute*). Příklad:

marise:~> ls -l To_Do -rw-rw-r--1 marise users 5 Jan 15 12:39 To_Do marise:~> ls -l /bin/ls -rwxr-xr-x 1 root root 45948 Aug 9 15:01 /bin/ls*

První soubor je obyčejný soubor (první znak je pomlčka). Jeho vlastníkem je uživatel *marise*, vlast-nickou skupinou je skupina *users*. Uživatel *marise* a členové skupiny *users* mohou soubor čísta zapisovat (měnit, přesunout, smazat), nemohou jej ale spustit (pro uživatele i skupinu jsou nasta-vena práva r a w, právo x chybí). Ostatní uživatelé mohou soubor pouze číst, nemohou jej pře-psat ani spustit (mají pouze právo r).

Druhý příklad ukazuje spustitelný soubor – všichni uživatelé mají právo jej spustit, přepsat jej však

může pouze uživatel *root*.Podrobnější popis způsobu zobrazení přístupových práv příkazem ls naleznete v informačníchstránkách, hledejte část *What information is listed*.
Pro snazší manipulaci mají jak přístupová práva (režimy), tak i kategorie uživatelů přiděleny své kódy, které najdete v následujících tabulkách.

Kód Význam

0 nebo -Odpovídající kategorii uživatelů se právo nepřiděluje. 4 nebo Odpovídající kategorii uživatelů se přiděluje právo čtení. 2 nebo Odpovídající kategorii uživatelů se přiděluje právo zápisu. 1 nebo Odpovídající kategorii uživatelů se přiděluje právo spuštění.

Kódy přístupových práv

Kód Význam

u Práva vlastníka souboru g Práva vlastnické skupiny o Práva ostatních uživatelů

Kódy kategorií uživatelů

Toto přímočaré schéma se striktně aplikuje na veškeré soubory, což umožňuje vysokou míru zabezpečení i bez dalších nástrojů síťové bezpečnosti. Mimo jiné se tímto schématem řídí přístup uživatelů k programům, umožňuje nastavovat přístup k souborům podle potřeb a chránit citlivá data, jako jsou domovské adresáře nebo konfigurační soubory.

Měli byste vědět, jaké je vaše uživatelské jméno. Pokud to náhodou nevíte, zjistíte je příkazem id, který vám rovněž sdělí, jaká je vaše výchozí skupina a kterých jiných skupin jste případně členem:

```
tilly:~> iduid=504(tilly) gid=504(tilly) groups=504(tilly),100(users),2051(org)
```

Vaše uživatelské jméno je uloženo také v proměnné USER:

```
tilly:~> echo $USER tilly
```

Nástroje Příkaz chmod

Logickým a občas obtížným důsledkem striktního systému přístupových práv je, že z řady různých důvodů bývá nutné přístupová práva změnit. Slouží k tomu příkaz chmod a v technické anglič-tině se dnes běžně používá sloveso *to chmod*, změnit přístupová práva. Příkaz chmod lze použít s textovými nebo číselnými parametry podle toho, co vám více vyhovuje.

Následující příkaz používá textové parametry a řeší poměrně obvyklý problém začínajících uživatelů:

```
asim:~> ./hello bash: ./hello: bad interpreter: Permission denied

asim:~> cat hello #!/bin/bash echo "Hello, World"

asim:~> ls -l hello
-rw-rw-r--1 asim asim 32 Jan 15 16:29 hello

asim:~> chmod u+x hello

asim:~> ./helloHello, World

asim:~> ls -l hello-rwxrw-r--1 asim asim 32 Jan 15 16:29 hello*
```

Operátory + a -slouží k přidání, respektive odebrání, příslušného práva. Užitečné příklady nalez-nete na informačních a manuálových stránkách. Následující příklad nastaví přístupová práva sou-boru z předchozího příkladu tak, že k němu bude mít přístup jen uživatel *asim*:

```
asim:~> chmod u+rwx,go-rwx hello

asim:~> ls -l hello-rwx------1 asim asim 32 Jan 15 16:29 hello*
```

Většina problémů, které se projevují chybovým hlášením „permision denied“, je způsobena nasta-vením přístupových práv. Obvyklé stížnosti „včera to ještě fungovalo“ a „když jsem *root*, tak to funguje“ mívají velmi často příčinu rovněž v nevhodně nastavených právech.

Při použití příkazu chmod s číselnými parametry se hodnoty práv pro každou kategorii uživate-lů sčítají. Dostanete tak trojici číslic udávající přístupová práva, která má chmod souboru nasta-vit. Následující tabulka ukazuje běžné kombinace nastavených práv:

Příkaz Význam

chmod 400 *soubor* Ochrana před neúmyslným smazáním souboru. chmod 500 *adresář* Ochrana před neúmyslným smazáním, přejmenováním nebo přesunutím adresáře. chmod 600 *soubor* Privátní soubor, který může měnit pouze jeho vlastník. chmod 644 *soubor* Veřejně čitelný soubor, který může měnit pouze jeho vlastník. chmod 660 *soubor* Soubor může číst a měnit vlastník a členové vlastnické skupiny, ostatní uživatelé nemají k sou-
boru přístup.

chmod 700 *adresář* Vlastník má k souboru plná práva, ostatní uživatelé žádná.

chmod 755 *soubor* Soubor mohou číst a spouštět všichni uživatelé, měnit jej však může jenom vlastník.

chmod 775 *soubor* Standardní nastavení sdílení souboru v rámci skupiny.

chmod 777 *soubor* Všichni mohou dělat se souborem všechno.

Příklady nastavení přístupových práv

Pokud jako hodnotu nastavovaných práv zadáte méně než tři číslice, chybějící údaje budou zleva doplněny nulami. Ve skutečnosti Linux pracuje ještě se čtvrtou číslicí, která předchází nám známé tři a nastavuje speciální přístupové režimy. Tyto a řadu dalších

informací naleznete na informačních stránkách.

Přihlášení do jiné skupiny

Když zadáte příkaz id, získáte seznam všech skupin, do nichž můžete potenciálně patřit. Kromě toho se zobrazí vaše uživatelské jméno a ID a jméno a ID skupiny, do níž právě patříte. Na řadě linuxových systémů můžete být v jednom okamžiku členem pouze jedné skupiny. Tato *primární skupina* se nastavuje v souboru /etc/passwd. Čtvrtý údaj v tomto souboru obsahuje právě ID primární skupiny, jejíž název následně můžete najít v souboru /etc/group. Příklad:

asim:~> iduid=501(asim) gid=501(asim) groups=100(users),501(asim),3400(web)

asim:~> grep *asim* /etc/passwdasim:x:501:501:Asim El Baraka:/home/asim:/bin/bash

asim:~> grep *501* /etc/groupasim:x:501:

Čtvrtá položka na řádku v souboru /etc/passwd obsahuje hodnotu „501“, která odpovídá skupině s názvem *asim*, jak můžeme vidět v souboru /etc/group. Po přihlášení do systému bude uživatel *asim* členem právě skupiny *asim*.

> Mechanismus privátních skupin
>
> Kvůli větší flexibilitě používá většina linuxových systémů mechanismus takzvaných *privát-ních skupin*, který každého uživatele zařazuje do jeho vlastní skupiny, proto „privátní sku-pina“. Obvykle se skupina jmenuje stejně jako uživatel, což může být občas trochu nepře-hledné.

Kromě této své primární skupiny může být uživatel *asim* ještě členem skupin *users* a *web*. Proto-že jde o jeho sekundární skupiny, musí se do nich explicitně přihlásit příkazem newgrp. V násle-dujícím příkladu vytváří uživatel *asim* soubor vlastněný skupinou *web*:

asim:/var/www/html> newgrp *web*

asim:/var/www/html> iduid=501(asim) gid=3400(web) groups=100(users),501(asim),3400(web)

Když nyní *asim* vytvoří soubor, jeho vlastnickou skupinou bude skupina *web*, nikoliv skupina *asim*:

asim:/var/www/html> touch test

asim:/var/www/html> ls -l test -rw-rw-r--1 asim web 0 Jun 10 15:38 test

Explicitním přihlášením do potřebné skupiny si ušetříte nutnost použít příkaz chown (viz kapi-tolu „Změna vlastníka a vlastnické skupiny“) nebo žádat administrátora, aby vlastnictví souboru změnil za vás.

Více informací naleznete na manuálové stránce příkazu newgrp.

Souborová maska

Když vytvoříte a uložíte nový soubor, budou mu nastavena výchozí přístupová práva, protože soubor „bez přístupových práv“ nemůže v Linuxu existovat. Výchozí práva jsou dána *maskou*, používanou při vytváření souborů. Hodnotu této masky zjistíte příkazem umask:

bert:~> umask
Na rozdíl od příkazu chmod, kde se symbolické hodnoty přístupových práv sčítají, při vytváření nového souboru se práva specifikovaná maskou odečítají od maximální množiny práv. V před-chozím příkladu vidíme čtveřici číselných hodnot, přestože uživatelské kategorie jsou pouze tři – vlastník, vlastnická skupina a ostatní. První nula se vztahuje k nastavení speciálních přístupových práv, o kterých budeme hovořit v kapitolách „Změna vlastníka a vlastnické skupiny“ a „SUID a SGID“. Je dokonce možné, že váš systém nebude úvodní nulu vypisovat a maska bude zobra-zena jako tříciferná hodnota.

Každý unixový systém má systémové volání pro vytvoření nového souboru, které se použije pokaždé, když uživatel vytváří soubor – například při stažení souboru z Internetu, při ukládání nového dokumentu a podobně. Tato funkce slouží k vytváření nových souborů i adresářů. V oka-mžiku vytvoření nového adresáře mu funkce nastaví plná práva pro všechny uživatele, při vytvá-ření nového souboru mu nastaví pro všechny uživatele práva číst a zapisovat, nenastaví však právo spouštění. Platí tedy, že ještě před aplikací souborové masky má nově vytvořený adresář práva *777* neboli *rwxrwxrwx* a nově vytvořený soubor *666* neboli *rw-rw-rw-*.

Jakmile dojde k vytvoření nového souboru či adresáře, od těchto plných práv se odečte hodnota masky. Jestliže má tedy maska hodnotu *(0)002*, pak nové adresáře budou mít práva *775* a nové soubory *664*. Ukazuje to následující příklad:

bert:~> mkdir newdir

bert:~> ls -ld newdirdrwxrwxr-x 2 bert bert 4096 Feb 28 13:45 newdir/

bert:~> touch newfile

bert:~> ls -l newfile -rw-rw-r--1 bert bert 0 Feb 28 13:52 newfile

Jestliže se příkazem newgrp přepnete do jiné skupiny, maska zůstane nezměněna. Pokud má tedy maska hodnotu *002*, pak soubory a adresáře, které vytvoříte „v rámci“ této nové skupiny, budou přístupné i pro ostatní členy skupiny, nebudete muset jejich práva měnit příkazem chmod.

Uživatel *root* má obvykle masku nastavenu striktněji:

[root@estoban root]# umask 022

Výchozí hodnota souborové masky je pro celý systém nastavena v konfiguračním souboru pro-středků shellu, například /etc/bashrc nebo /etc/profile. Můžete ji změnit ve svém vlastním konfi-guračním souboru shellu, viz kapitolu „Home, sweet /home“, kde hovoříme o vlastním nastavení shellu.

Změna vlastníka a vlastnické skupiny

Pokud soubor vlastní nesprávný uživatel nebo skupina, je to možné změnit příkazem chown (*change owner*) nebo chgrp (*change group*). V prostředích, kde se soubory sdílejí v rámci sku-pin, je změna vlastnictví poměrně častou administrativní operací.

Oba příkazy jsou značně pružné, podrobnosti můžete zjistit tak, že je spustíte s přepínačem --help. Prostřednictvím příkazu chown je možno současně změnit jak vlastníka, tak vlastnickou skupinu,
příkazem chgrp je možno změnit pouze skupiny. Systém samozřejmě ověřuje, zda uživatel, kterýchce změnu provést, má k této operaci odpovídající práva.Chcete-li změnit pouze vlastníka souboru, použijte následující příkaz:
chown *vlastník soubor*

Pokud bezprostředně za jménem nového vlastníka zadáte dvojtečku (viz informační stránky), změní se i vlastnická skupina na primární skupinu nového vlastníka. Na většině linuxových systémů má každý uživatel svou vlastní privátní primární skupinu, takže tímto způsobem lze daný soubor nastavit jako privátní soubor uživatele:

jacky:~> iduid=1304(jacky) gid=(1304) groups=1304(jacky),2034(pproject)

jacky:~> ls -l my_report-rw-rw-r--1 jacky project 29387 Jan 15 09:34 my_report

jacky:~> chown jacky: my_report

jacky:~> chmod o-r my_report

jacky:~> ls -l my_report -rw-rw----1 jacky jacky 29387 Jan 15 09:34 my_report

Bude-li chtít uživatel sdílet soubor s dalšími uživateli v nějaké skupině, může použít příkaz chgrp:

jacky:~> ls -l report-20020115.xls -rw-rw---- 1 jacky jacky 45635 Jan 15 09:35 report-20020115.xls

jacky:~> chgrp project report-20020115.xls

jacky:~> chmod o= report-20020115.xls

jacky:~> ls -l report-20020115.xls -rw-rw---- 1 jacky project 45635 Jan 15 09:35 report-20020115.xls

Nyní budou moci se souborem pracovat všichni členové skupiny *project*. Uživatelé, kteří nejsou
členem této skupiny, nebudou mít k souboru žádný přístup.Příkazy chown i chgrp je možné pomocí přepínače -R použít rekurzivně. V takovém případědojde ke změně vlastnictví všech souborů a podadresářů v daném adresáři.

Omezení

Na většině systémů mají běžní uživatelé z bezpečnostních důvodů zakázáno příkazy chown a chgrp používat. Zlovolný uživatel by totiž mohl nastavovat vlastnictví souborů na jiné uživatele a skupiny, měnit tak chování prostředí těchto uživatelů a v krajním pří-padě dokonce poškodit jejich soubory.

Speciální přístupová práva

Aby správce systému nemusel trvale řešit různé problémy s přístupovými právy, je možné adre-sářům nebo konkrétním programům nastavovat speciální přístupová práva. Jde o následující troji-ci práv:

- Sticky bit: Původní význam tohoto příznaku byl takový, že po skončení úlohy zůstal pří-slušný příkaz v systémové paměti. Používal se k šetření pamětí – velké úlohy se nahráva-ly do paměti jen jednou. V současné době je ovšem paměť levná a mechanismy její sprá-vy se výrazně vylepšily, takže optimalizace chování jednotlivých souborů pomocí tohoto příznaku už není zapotřebí. Pokud ovšem příznak nastavíte celému adresáři, jeho význam bude jiný. V takovém případě bude moci uživatel měnit soubory v tomto adresáři pouze v případě, že je vlastníkem příslušného souboru nebo má soubor nastavena příslušná práva. Této funkce se používá v adresářích jako /var/tmp, do nichž musí mít právo zápi-su všichni uživatelé, není ale žádoucí, aby jeden uživatel mohl mazat soubory vytvořené jiným uživatelem. Nastavení sticky bitu je indikováno písmenem *t* na posledním místě výpi-su práv:

  mark:~> ls -ld /var/tmpdrwxrwxrwt 19 root root 8192 Jan 16 10:37 /var/tmp/

Příznak se nastavuje příkazem chmod o+t *adresář*. Značka „t“ historicky pochází z Unixu, její význam byl *save Text access*.

Příznaky SUID (*set user ID*) a SGID (*set group ID*): Indikuje je znak *s* v poli práv vlastníka nebo vlastnické skupiny. Pokud jsou tyto příznaky nastaveny u spustitelného souboru, pak soubor po spuštění poběží s právy vlastníka a/nebo vlastnické skupiny, nikoliv s právy uži-vatele, který soubor spustil. Tímto mechanismem je možné i neprivilegovaným uživatelům poskytnout přístup k jinak nepřístupným systémovým prostředkům. Více viz kapitolu „Pro-cesy“
Příznak SGID (*set group ID*) nastavený u adresáře: V tomto speciálním případě bude mít každý v daném adresáři nově vytvořený soubor nastavenu vlastnickou skupinu na tu, která vlastní příslušný adresář. (Standardní chování je, že vlastnickou skupinou bude skupina, jíž je vytvářející uživatel právě členem.) Díky tomu se uživatel ve sdílených adresářích nemu-sí zabývat vlastnictvím souborů:

mimi:~> ls -ld /opt/docsdrwxrws---4 root users 4096 Jul 25 2001 docs/

mimi:~> ls -l /opt/docs -rw-rw----1 mimi users 345672 Aug 30 2001-Council.doc

Tento mechanismus se v Unixu používá standardně při sdílení souborů.

U existujících souborů se vlastnictví nemění

Pokud do SGID adresáře přesunete soubor původně vytvořený na jiném místě, zachová se jeho původní vlastník a vlastnická skupina. To může občas vést k omylům.

Shrnutí

V Linuxu, stejně jako v Unixu, jsou všechny systémové entity systému tak či onak prezentovány jako soubory s různými vlastnostmi. Díky (předdefinované) cestě mohou uživatelé snadno nalézt, číst a spouštět soubory.

Máme za sebou první krok na cestě, jejímž cílem je stát se expertem. Hovořili jsme o vnějším vzhledu i interním uspořádání souborového systému a známe souborový bezpečnostní model Linuxu a některá další bezpečnostní opatření, která systém standardně používá.

Nejdůležitějším nástrojem pro práci se systémem je shell. V této kapitole jsme se naučili používat několik příkazů shellu, jejich souhrn uvádí následující tabulka.

Příkaz	Popis
bash	Spustí GNU shell.
cat *soubor(y)*	Pošle obsah souboru na standardní výstup.
cd *adresář*	Přejde do zadaného adresáře, jde o vestavěný příkaz shellu.
chgrp *nová_skupina soubor(y)*	Změní vlastnickou skupinu souboru.
chmod *práva soubor(y)*	Změní přístupová práva souboru.
chown *nový_vlastník*[: [*nová_skupina*]] *soubor(y)*	Změní vlastníka a případně vlastnickou skupinu souboru.
cp *zdrojový_soubor cílový_soubor*	Zkopíruje zdrojový soubor na cílový soubor.
df [*soubor*]	Vypíše informace o využití diskového oddílu, na němž se nachází soubor.
echo *řetězec*	Vypíše řetězec.
export	Součást shellu, propaguje do systému proměnné a jejich hodnoty.
file *soubor*	Zjistí typ souboru.
find *cesta výraz*	Hledá soubor podle názvu.

grep *řetězec soubor*	Zobrazuje řádky souboru, které obsahují zadaný řetězec.
head *soubor*	Vypíše začátek souboru.
id	Vypíše reálné a efektivní identifikátory uživatele a skupiny.
info *příkaz*	Zobrazí informační stránky příkazu.
less *soubor*	Spustí prohlížeč souborů.
ln *cíl_odkazu název_odkazu*	Vytvoří odkaz na zadaný cíl.
locate *výraz*	Hledá soubor podle názvu.
ls *soubor(y)*	Vypíše obsah adresáře.
man *příkaz*	Vypíše manuálovou stránku příkazu.
mkdir *adresář*	Vytvoří nový adresář.
mv *starý_název nový_název*	Přesune nebo přejmenuje soubor nebo adresář.
pwd	Vypíše aktuální adresář.
quota	Vypíše informace o využití disku a nastavených limitech.
rm *soubor*	Odstraní soubor.
rmdir *adresář*	Odstraní adresář.
tail *soubor*	Vypíše konec souboru.
umask [*hodnota*]	Zobrazí nebo nastaví implicitní masku práv pro vytváření souborů.
wc *soubor*	Spočítá řádky, slova a znaky v souboru.
which *příkaz*	Vypíše cestu k příkazu.

Nové příkazy

Dále jsme zdůrazňovali nezbytnost číst manuálové stránky. Tato dokumentace je vaším primárním pomocníkem a obsahuje odpovědi na řadu otázek. V předcházející tabulce je uveden seznam základních příkazů, které budete denně používat. Tyto příkazy dokážou mnohem více než jen to, o čem jsme v této kapitole hovořili. V dokumentaci naleznete další možnosti využití těchto příkazů.

A nakonec přehled přístupových práv:

Kdo/Co čtení (r) zápis (w) spuštění (x)

uživatel (u)	2	1	4
skupina (g)	2	1	4
ostatní (o)	4	2	1

Přístupová práva souborů

Cvičení

Přihlaste se pod svou normální uživatelskou identitou.

Diskové oddíly

Na kterém diskovém oddílu máte domovský adresář?
Kolik je v systému diskových oddílů?
Jaká je celková velikost nainstalovaného systému?

Cesty

Vypište vyhledávací cestu
Vyexportujte nesmyslnou cestu, například příkazem export PATH=blabla, a zkuste vypsat obsah adresáře
Jaká je cesta k vašemu domovskému adresáři? Jakou relativní cestu musí použít jiný uživatel, aby se ze svého domovského adresáře dostal do vašeho?

Přejděte do podadresáře tmp v adresáři /var
Nyní jediným příkazem přejděte do podadresáře share v adresáři /usr. Změňte adresář na doc. Jaký je váš aktuální adresář?

Prohlídka systému

Přejděte do adresáře /proc
Na jakém procesoru váš systém běží?
Kolik paměti systém právě používá?
Kolik máte odkládacího prostoru?
Které ovladače jsou zavedeny?
Jak dlouho systém běží?
Jaké souborové systémy váš systém zná?
Přejděte do adresáře /etc/rc.d, /etc/init.d nebo /etc/runlevels a zvolte adresář odpovídající aktuální úrovni běhu
Jaké služby by měly v této úrovni běžet?
Které služby běží v grafickém režimu a neběží v textovém?
Přejděte do adresáře /etc
Jak dlouho uchovává systém soubor se záznamy o přihlášení uživatelů?
Jakou verzi systému používáte?
Máte nastavenu nějakou zprávu dne?
Kolik uživatelů je v systému vytvořeno? Nepočítejte je, ať je spočítá systém!
Kolik skupin?
Kde jsou uloženy informace o časovém pásmu?
Přejděte do adresáře /use/share/doc.
Jsou v systému nainstalovány nějaké dokumenty HOWTO?
Vyjmenujte tři programy, které jsou součástí GNU balíku *coreutils*.
Jakou verzi bashe váš systém používá?

Manipulace se soubory

Ve svém domovském adresáři vytvořte nový podadresář.
Můžete adresář přesunout na stejnou úroveň, na níž se nachází váš domovský adresář?
Do tohoto nového adresáře zkopírujte všechny soubory XPM z adresáře /usr/share/pixmaps. Co to jsou soubory XPM?
Vypište soubory sestupně seřazené podle abecedy.
Přejděte do domovského adresáře. Vytvořte nový adresář a zkopírujte do něj všechny sou-bory z adresáře /etc. Zkopírujte i všechny adresáře a podadresáře v tomto adresáři (rekur-zivní kopírování)!
Přejděte do tohoto adresáře a vytvořte v něm podadresář pro soubory, jejichž název začí-ná velkým písmenem, a jiný pro soubory, jejichž název začíná malým písmenem. Přesuň-te soubory do odpovídajících adresářů. Použijte co nejmenší množství příkazů.
Smažte zbývající soubory.
Jediným příkazem smažte celý adresář i s obsahem.
Pomocí příkazu grep najděte skript, který v grafickém režimu spustí Font Server.
Kde najdete program sendmail?
Vytvořte ve svém adresáři symbolický odkaz na adresář /var/tmp. Ověřte, zda funguje.
Vytvořte ve svém adresáři další symbolický odkaz, vedoucí na dříve vytvořený odkaz. Ověřte, zda funguje. Odstraňte první odkaz a vypište obsah adresáře. Co se stalo s druhým odkazem?

Přístupová práva

Můžete změnit práva adresáře /home?
Jaký je standardní režim pro vytváření souborů?
Změňte vlastníka a vlastnickou skupinu adresáře /etc na sebe a svou skupinu.
Změňte práva souboru ~/.bashrc tak, abyste jej mohli číst pouze vy a vaše primární skupina.
Zadejte příkaz locate root. Stane se něco zajímavého?
Vytvořte symbolický odkaz na adresář /root. Půjde použít?

Procesy

Vedle souborů jsou další důležitou věcí v unixových a linuxových systémech procesy. V této kapi-tole se budeme procesy zabývat podrobněji. Dozvíme se více o následujících tématech:

Víceuživatelské a víceúlohové prostředí.
Typy procesů
Ovládání procesů pomocí různých signálů
Atributy procesů
Životní cyklus procesu
Spuštění a zastavení systému
SUID a SGID
Rychlost a reakční doba systému
Plánování procesů
Systém pro periodické spouštění úloh – Vixie cron
Jak ze systému dostat maximum

Podrobně o procesech

Víceuživatelské a víceúlohové prostředí

Už jsme si trochu zvykli na prostředí operačního systému a dokážeme s ním komunikovat, takže je čas podrobněji se seznámit s procesy, které spouštíme. Ne každý příkaz spustí jeden proces. Některé příkazy, například mozilla, spustí více procesů najednou, jiné, například ls, spustí sku-tečně jen jeden proces.

Linux je založen na filozofii Unixu, kde je běžné, že více uživatelů spouští více příkazů, a to ve stejnou chvíli a na stejném systému. Je jasné, že musí existovat nějaký mechanismus, díky němuž bude procesor všechny tyto procesy obsluhovat a který uživatelům umožní přepínat se mezi pro-cesy. V některých případech může zůstat proces spuštěný i poté, co se uživatel, který jej spustil, odhlásí. Uživatel musí mít také možnost znovu aktivovat přerušené procesy.

V následujících částech se budeme věnovat struktuře fungování procesů v Linuxu.

Typy procesů Interaktivní procesy

Interaktivní procesy jsou spouštěny a ovládány prostřednictvím terminálové relace. Jinak řečeno, někdo musí být v systému přihlášen a proces spustit, nespouštějí se samy jako součást systému. Interaktivní proces může běžet v popředí, pak obsadí terminálové okno, v němž byl spuštěn,
a dokud proces na popředí neskončí, nemůžete z okna spustit jiný proces. Doposud jsme téměřvýhradně pracovali s procesy, které běžely na popředí – doba jejich běhu ovšem byla příliš krátká,než aby to bylo možno postřehnout. Dobrým příkladem ale může být příkaz less, který obsadí oknoa nedovolí vám v něm dělat něco jiného. V tomto konkrétním případě příkaz čeká, až něco udělá-te. Program je svázán s terminálem, z nějž byl spuštěn, a terminál můžete použít pouze k zadáváníkazů, jimž program rozumí. Jiné příkazy vyvolají chybu nebo je program bude ignorovat.

Pokud ovšem proces běží na pozadí, uživateli nic nebrání v použití terminálu, z nějž běžící pro

ces spustil.Shell má vestavěny funkce pro *řízení úloh*, díky nimž je možno snadno manipulovat s více pro-cesy. Pomocí těchto mechanismů je možno procesy přepínat mezi popředím a pozadím, případ-ně je možné rovnou spustit proces na pozadí.

Spouštění procesu na pozadí má smysl pouze v případě, že proces nevyžaduje uživatelský vstup(prostřednictvím shellu). Typicky se na pozadí nechávají běžet úlohy, u nichž se předpokládá delšíčas trvání. Proces spustíte na pozadí (a terminálové okno si tak uvolníte ihned po jeho spuštění)tak, že za příkazem uvedete znak &. V následujícím příkladu v grafickém režimu otevřeme novéterminálové okno ze stávajícího okna:

```
billy:~> xterm &
[1] 26558

billy:~> jobs[1]+ Running xterm &
```

Podrobně je mechanismus řízení úloh popsán na informačních stránkách bashe, proto si nyní uvedeme jen základní přehled funkcí pro řízení úloh:

Operace Popis

příkaz Spustí příkaz na popředí. *příkaz* & Spustí příkaz na pozadí. jobs Vypíše seznam úloh na pozadí. Ctrl+Z Pozastaví (tedy zastaví běh, ale neukončí) úlohu běžící na popředí. Ctrl+C Zastaví (ukončí) úlohu běžící na popředí. %*n* Každý proces běžící na pozadí má přiřazeno své číslo. Zápisem %n je možno se na proces na
pozadí odkazovat jeho číslem, například .

bg Reaktivuje pozastavenou úlohu na pozadí.

fg Přesune úlohu z pozadí na popředí.

kill Ukončí proces. (Viz Shell Builtin Commands na informačních stránkách bashe.)

Řízení procesů

Další praktické příklady naleznete ve cvičeních. Většina unixových systémů umožňuje použít příkaz screen, který je užitečný v situaci, kdy chce-te příkaz spustit v jiné instanci shellu. Když spustíte příkaz screen, vytvoří se nová relace se svým vlastním shellem a/nebo zadanými příkazy, kterou můžete následně opustit. V nové relaci může-te dělat cokoliv, co potřebujete. Všechny programy a činnosti běží nezávisle na původním shellu. Když se od relace odpojíte, spuštěné programy i nadále běží, dokonce i když se odhlásíte z původního shellu. Kdykoliv později se pak můžete k běžící relaci vrátit.
Tento program pochází z dob, kdy neexistovaly virtuální konzoly a všechno bylo nutné dělat z jediného textového terminálu. Mechanismus však v Linuxu zůstal zachován, přestože už více než deset let používáme virtuální terminály. Příkaz screen dnes oceníte například v situaci, kdy chce-te spustit nějakou úlohu a později ji například zkontrolovat z jiného počítače – na počítači A spus-títe úlohu prostřednictvím příkazu screen a z počítače B se můžete později přihlásit k počítači A a vstoupit do úlohy, kterou jste na něm dříve spustili.

Automatické procesy

Automatické či dávkové procesy nejsou spojeny s terminálem. Takové procesy se řadí do fronty, v níž se obsluhují mechanismem FIFO (*first in, first out*). Spouštění úloh uložených ve frontě je možné na základě jednoho ze dvou kritérií:

V určitém okamžiku – k tomu slouží příkaz at, o němž budeme hovořit v další části této kapitoly.
Ve chvíli, kdy zatížení (load) systému poklesne pod určitou hranici – k tomu slouží příkaz batch. Při standardním nastavení se procesy začínají obsluhovat, pokud zatížení systému poklesne pod 0,8. Na větších systémech se tohoto mechanismu používá, pokud je nutné spustit úlohu, která zpracovává velké množství dat nebo má velkou spotřebu systémových prostředků. Tímto způsobem se optimalizuje výkon systému.

Démoni

Démoni jsou trvale běžící procesy serverů. Ve většině případů se spouštějí při startu systému a čekají na pozadí, dokud nedorazí požadavek na příslušnou službu. Typickým příkladem je síťo-vý démon *xinetd*, která se spouští téměř vždy při startu počítače. Po nabootování systému démon pouze sedí a čeká, dokud se nebude chtít připojit nějaký síťový klient.

Atributy procesů

Každý proces má svou množinu charakteristik:

Identifikátor procesu, PID – jedinečná číselná hodnota identifikující proces.
Identifikátor rodičovského procesu, PPID – PID procesu, který spustil daný proces.
Hodnotu *nice* – udává míru toho, jak je proces „hodný“ na ostatní procesy. (Nezaměňovat s prioritou procesu, která se počítá právě na základě hodnoty *nice* a předchozího využití procesoru procesem.)
Číslo terminálu, k němuž je proces připojen.
Identifikátory reálného a efektivního uživatele (RUID a EUID) – vlastník procesu. Reálný uživatel je ten, kdo proces spustil. Efektivní uživatel udává práva procesu k systémovým prostředkům. RUID a EUID jsou ve většině případů stejné a proces má stejná práva jako uživatel, který jej spustil.

Toto téma si zasluhuje trochu objasnění. Prohlížeč mozilla v adresáři /usr/bin je vlastněn uživa-telem *root*:

theo:~> ls -l /usr/bin/mozilla-rwxr-xr-x 1 root root 4996 Nov 20 18:28 /usr/bin/mozilla*

theo:~> mozilla &
[1] 26595

theo:~> ps -af UID PID PPID C STIME TTY TIME CMD theo 26601 26599 0 15:04 pts/5 00:00:00 /usr/lib/mozilla/mozilla-bin theo 26613 26569 0 15:04 pts/5 00:00:00 ps -af

Jakmile uživatel *theo* tento program spustí, příslušný proces a všechny jím dále vytvořené proce-sy budou vlastněny uživatelem *theo*, nikoliv uživatelem *root*. Pokud bude prohlížeč potřebovat pří-stup k nějakým souborům nebo adresářům, přístupová práva se budou vyhodnocovat pro uživa-tele *theo*.

- Identifikátory reálné a efektivní vlastnické skupiny (RGID a EGID) – reálná skupina je pri-mární skupina uživatele, který proces spustil. Efektivní skupina je obvykle stejná, leda by šlo o soubor s nastaveným příznakem SGID.

Zobrazení informací o procesech

Základním nástrojem pro vypsání spuštěných procesů je příkaz ps. Tento příkaz používá celouřadu přepínačů, jejichž kombinacemi lze vypisovat různé procesy a různé atributy.Při použití bez dalších voleb příkaz vypíše pouze aktuální shell a v

něm spuštěné procesy:

```
theo:~> ps
   PID TTY TIME CMD
  4245 pts/7 00:00:00 bash
  5314 pts/7 00:00:00 ps
```

Takový výpis nám ale málokdy stačí, v průměrném systému totiž běží klidně i stovky procesů. Vět-šinou se proto příkaz ps používá v kombinaci s *rourou* (viz kapitolu „Přesměrování výstupu ope-rátory > a “) a příkazem grep, jímž z úplného seznamu procesů vyfiltrujeme ty, které nás zajíma-jí. Následujícím příkazem bychom například zobrazili všechny procesy vlastněné určitým uživate-lem:

```
ps -ef | grep uživatel
```

Další příklad ukazuje výpis procesů s názvem bash, tedy nejběžnějšího linuxového shellu:

```
theo:> ps auxw | grep bash brenda 31970 0.0 0.3 6080 1556 tty2 S Feb23 0:00 -bash root 32043 0.0 0.3 6112 1600 tty4 S Feb23 0:00 -bash theo 32581 0.0 0.3 6384 1864 pts/1 S Feb23 0:00 bash

theo 5427 0.0 0.1 3720 548 pts/7 S 19:22 0:00 grep bash
```

V takovýchto případech se ve výpisu procesů může objevit i samotný příkaz grep, který jsme pou-žili k přefiltrování výpisu. Pokud by vám to vadilo, použijte místo něj příkaz pgrep nebo zkuste filtrovat výpis procesů takto: grep [b]ash

Shell je poněkud speciální případ – v seznamu procesů můžeme rozlišit, které shelly byly spuště-ny jako přihlašovací (ve kterých jste zadávali jméno a heslo, buď při textovém nebo vzdáleném přihlášení) a které přihlašovací nejsou (byly spuštěny například v terminálovém okně). U přihla-šovacích shellů předchází názvu pomlčka.

Další informace získáte obvyklým způsobem: ps --help nebo man ps. GNU verze příkazu ps nabízí celou řadu voleb pro formátování výpisu. Příkaz ps zobrazuje pouze okamžitý stav aktivních procesů, jde o jednorázový snímek. Příkazem top můžete získat seznam procesů, který se každých pět sekund aktualizuje a je seřazen podle toho, jak jednotlivé procesy zatěžují systém. Navíc příkaz poskytuje i několik celkových statistik o využití systému:

```
  12:40pm up 9 days, 6:00, 4 users, load average: 0.21, 0.11, 0.03 89 processes: 86 sleeping, 3 running, 0 zombie, 0 stopped CPU states: 2.5% user, 1.7% system, 0.0% nice, 95.6% idle Mem: 255120K av, 239412K used, 15708K free, 756K shrd, 22620K buff Swap: 1050176K av, 76428K used, 973748K free, 82756K cached

   PID USER PRI NI SIZE RSS SHARE STAT %CPU %MEM TIME COMMAND
  5005 root 14 0 91572 15M 11580 R 1.9 6.0 7:53 X 19599 jeff 14 0 1024 1024 796 R 1.1 0.4 0:01 top 19100 jeff 9 0 5288 4948 3888 R 0.5 1.9 0:24 gnome-terminal 19328 jeff 9 0 37884 36M 14724 S 0.5 14.8 1:30 mozilla-bin
     1 root 8 0 516 472 464 S 0.0 0.1 0:06 init
     2 root 9 0 0 0 0 SW 0.0 0.0 0:02 keventd
     3 root 9 0 0 0 0 SW 0.0 0.0 0:00 kapm-idled
     4 root 19 19 0 0 0 SWN 0.0 0.0 0:00 ksoftirqd_CPU0
     5 root 9 0 0 0 0 SW 0.0 0.0 0:33 kswapd
     6 root 9 0 0 0 0 SW 0.0 0.0 0:00 kreclaimd
     7 root 9 0 0 0 0 SW 0.0 0.0 0:00 bdflush
     8 root 9 0 0 0 0 SW 0.0 0.0 0:05 kupdated
     9 root -1-20 0 0 0 SW< 0.0 0.0 0:00 mdrecoveryd
    13 root 9 0 0 0 0 SW 0.0 0.0 0:01 kjournald
    89 root 9 0 0 0 0 SW 0.0 0.0 0:00 khubd
   219 root 9 0 0 0 0 SW 0.0 0.0 0:00 kjournald
   220 root 9 0 0 0 0 SW 0.0 0.0 0:00 kjournald
```

Na prvním řádku výpisu příkazu top vidíte stejné údaje, které poskytuje příkaz uptime:

```
jeff:~> uptime 3:30pm, up 12 days, 23:29, 6 users, load average: 0.01, 0.02, 0.00
```

Údaje, které tyto programy vypisují, jsou mimo jiné uloženy v souboru /var/run/utmp (seznam aktuálně přihlášených uživatelů) a ve virtuálním souborovém systému /proc, například v souboru /proc/loadavg (informace o průměrném zatížení systému). Stejné údaje lze zobrazit i celou řadou grafických nástrojů, například programem *Gnome System Monitor* nebo *lavaps*. Na serverech FreshMeat (*http://www.freshmeat.net*) nebo SourceForge (*http://www.sourceforge.org*) naleznete celou řadu aplikací, které tyto informace centrálně shromažďují a zobrazují, dokážou je kombino-vat s dalšími údaji z jiných počítačů a umožňují tak z jednoho místa monitorovat celou infra-strukturu.

Vzájemné vztahy mezi procesy ukazuje příkaz pstree:

```
sophie:~> pstree
init-+-amd
     |-apmd
     |-2*[artsd]
     |-atd
     |-crond
     |-deskguide_apple
     |-eth0
     |-gdm---gdm-+-X
     | `-gnome-session-+-Gnome
     | |-ssh-agent | `-true |-geyes_applet |-gkb_applet |-gnome-name-serv |-gnome-smproxy |-gnome-terminal-+-bash---vim | |-bash | |-bash---
     pstree | |-bash---ssh | |-bash---mozilla-bin---mozilla-bin---3*[mozilla-bin] | `-gnome-pty-helper <zkráceno>
```

Další informace můžete zobrazit pomocí přepínačů -u a -a. Seznam všech přepínačů a jejich funk-ci naleznete na informačních stránkách.V další části si ukážeme, jak může jeden proces vytvořit jiný proces.

Vznik a zánik procesů Vytvoření procesu

Nový proces vzniká tak, že existující proces vytvoří svou přesnou kopii. Tento synovský proces má prostředí nastaveno stejně jako jeho rodič, liší se od něj pouze svým identifikačním číslem. Této operaci se říká *forking*.

Po vytvoření nového procesu se jeho adresní prostor přepíše daty nového procesu. To se prová dí systémovým voláním *exec*. Tento mechanismus *fork-a-exec* tak nahradí starý příkaz novým, přičemž prostředí, v němž nový program běží, zůstává zachováno, a to včetně konfigurace vstupních a výstupních zařízení, pro-měnných prostředí a priority. Tento mechanismus se používá k vytváření všech unixových proce-sů, a platí tedy i v Linuxu. Dokonce i úplně první proces, init, s číslem 1 vznikl forkem při zavá-dění systému v zaváděcí proceduře.
Následující schéma ilustruje činnost mechanismu *fork-a-exec*. S každým forkem se mění identifi-kátor procesu. Mohou nastat situace, kdy se rodičem nějakého procesu stane proces init i přesto, že daný pro-ces nebyl původně procesem init spuštěn. Příklad takových procesů jsme mohli vidět výše ve výpisu příkazu pstree. Například některé programy *démonizují* své potomky, takže potomek může běžet i poté, co rodičovský proces skončí. Typickým příkladem je správce oken – spustí program xterm, v němž poběží shell, ve kterém můžete zadávat příkazy. Správce oken se ovšem vzdává jakékoliv zodpovědnosti a předává synovský proces procesu init. Díky tomuto mechanis-mu můžete správce oken změnit, aniž by to ovlivnilo běžící aplikace.

Dokonce i v nejlepších rodinách se čas od času něco pokazí. Ve výjimečných případech se stává, že proces skončí, aniž by rodič počkal na zprávu o jeho dokončení (což by správně udělat měl). Z takového „nepohřbeného“ procesu se stává tzv. z*ombie* – proces, který zůstane v systému „viset“.

Ukončení procesu

Pokud proces skončí normálně (nedojde k jeho zabití nebo jinému neočekávanému přerušení), vrátí svému rodičovskému procesu *návratový kód*. Tato číselná hodnota rodiči indikuje, jak běh potomka skončil. Mechanismus předávání zprávy o výsledku běhu úlohy má původ v jazyce C, v němž byl systém Unix naprogramován.
Rodičovský proces nebo volající skript pak může návratový kód potomka nějak interpretovat. Vý-znam předávané návratové hodnoty samozřejmě závisí na volaném programu. Možné návratové hodnoty jsou obvykle popsány na manuálových stránkách – například program grep vrátí hod-notu -1 v případě, že nenalezl žádný hledaný řádek. Jiným příkladem může být vestavěný příkaz true, který nedělá vůbec nic a pouze vrátí hodnotu 0, indikující úspěšné skončení.

Uvědomte si, že vztahy rodič a potomek mezi procesy sice názorně popisují, jak procesy vznika-jí, tuto analogii ale nelze brát úplně striktně. Biologický svět je založen na principu, že potomci ve většině případů přežívají své rodiče a sami se rodiči stávají. V unixovém světě je naopak základ-ní princip ten, že rodič vytvoří potomka a následně čeká, až potomek skončí, je to tedy přesně opačně než v životě. Ve specifických situacích, kdy rodič nemá v plánu svého potomka přežít, to musí nějak explicitně ošetřit (například svěřením potomka do péče procesu init), jinak by vznikl chaos.

Signály

Procesy mohou skončit tak, že dostanou příslušný signál. Procesu můžete poslat různé signály, používá se k tomu příkaz kill. Seznam signálů, které systém zná, můžete zjistit příkazem kill -l. Většinu signálů používá systém interně nebo je používají autoři programů k ošetření různých spe-cifických potřeb. Řadový uživatel obvykle vystačí s následujícími signály:

Název signálu Číslo signálu Význam

SIGTERM 15 Řádné ukončení procesu. SIGINT 2 Přerušení procesu. Proces může tento signál ignorovat. SIGKILL 9 Přerušení procesu. Proces nemůže tento signál ignorovat. SIGHUP 1 U démonů vynutí nové načtení konfigurace.

Běžně používané signály

Více informací o standardním chování procesů po přijetí různých signálů můžete získat na manu-álové stránce man 7 signal.

SUID a SGID

Jak jsme slíbili už v minulé kapitole, budeme nyní podrobněji hovořit o speciálních režimech SUID a SGID. Tyto režimy existují proto, aby normální uživatelé mohli provádět úkony, které by za nor-málních okolností nemohli provést v důsledku přísně nastavených přístupových práv unixového systému. V ideálním případě by se tyto speciální režimy měly používat co možná nejméně, pro-tože představují určité bezpečnostní riziko. Linux je navržen tak, aby se použití těchto režimů co možná nejvíce předešlo. Například linuxová verze příkazu ps vychází z informací dostupných v souborovém systému /proc, který je stejně přístupný komukoliv, a nehrozí tedy, že by mohlo dojít k úniku citlivých systémových dat. Naproti tomu starší unixové verze příkazu ps získávaly informace ze souborů /dev/mem a /dev/kmem, což je méně vhodné, protože tyto soubory nejsou normálním uživatelům přístupné:

rita:~> ls -l /dev/*mem crw-r-----1 root kmem 1, 2 Aug 30 22:30 /dev/kmem crw-r-----1 root kmem 1, 1 Aug 30 22:30 /dev/mem

Starší verzi programu ps tak normální uživatel nemohl spustit, pokud programu nebyl nastaven speciální přístupový režim. Jakkoliv obecně platí, že se použití speciálních přístupových režimů snažíme vyhnout, občas je jejich použití nezbytné. Příkladem může být mechanismus změny hesla. Je jasné, že uživatelé si budou chtít měnit heslo sami, nebudou chtít, ať jim je nastavuje správce systému. Jak víme, uži-vatelská jména a hesla jsou uložena v souboru /etc/passwd, který má následující přístupová práva a vlastnictví:

bea:~> ls -l /etc/passwd -rw-r--r--1 root root 1267 Jan 16 14:43 /etc/passwd

I tak ale uživatelé potřebují mít možnost měnit své údaje v tomto souboru. Proto má program passwd nastaven speciální režim SUID:

mia:~> which passwdpasswd is /usr/bin/passwd

mia:~> ls -l /usr/bin/passwd-r-s--x--x 1 root root 13476 Aug 7 06:03 /usr/bin/passwd*

Když program passwd spustíte, poběží s právy svého vlastníka, tedy uživatele *root*, takže i nor-mální uživatel bude moci změnit své heslo a změněné údaje do souboru /etc/passwd zapsat, přes-tože do tohoto souboru může zapisovat jen *root*.

SGID režim se u souborů používá méně často než SUID, protože při použití SGID příznaku bývá obvykle nutné vytvořit i zvláštní skupinu. V některých případech je ale takový postup nezbytný k elegantnímu vyřešení některých složitějších situací. (A nemusíte mít příliš obavy, protože potřeb-né skupiny bývají obvykle vytvořeny už při instalaci.) Příkladem mohou být programy write a wall, které posílají zprávy na terminály jiných uživatelů. Příkaz write pošle zprávu jednomu uži-vateli, příkaz wall napíše všem připojeným uživatelům.

Zapsat text na terminál či grafický displej jiného uživatele není normálně možné. Celý problém je vyřešen tak, že existuje zvláštní skupina, která vlastní všechna terminálová zařízení. Příkazy write a wall mají nastaven režim SGID, a běží tedy s právy své vlastnické skupiny, což je v tomto pří-
padě skupina *tty*. Tato skupina má právo zapisovat na cílové terminály, takže uživatel může jejím prostřednictvím poslat zprávu na libovolný terminál, což by jinak nebylo možné.V následujícím příkladu si uživatel *joe* nejprve příkazem who zjistí, ke kterému terminálu je při-pojen uživatel, jemuž chce napsat. Následně mu příkazem write pošle zprávu. Příklad také uka-zuje nastavení přístupových práv programu write a terminálu, k němuž je příjemce zprávy při-hlášen – vidíme, že k terminálu má plná přístupová práva pouze aktuálně přihlášený uživatel,navíc na něj může zapisovat vlastnická skupina.

joe:~> which writewrite is /usr/bin/write

joe:~> ls -l /usr/bin/write-rwxr-sr-x 1 root tty 8744 Dec 5 00:55 /usr/bin/write*

joe:~> whojenny tty1 Jan 23 11:41jenny pts/1 Jan 23 12:21 (:0)jenny pts/2 Jan 23 12:22 (:0)jenny
pts/3 Jan 23 12:22 (:0)joe pts/0 Jan 20 10:13 (lo.callhost.org)

joe:~> ls -l /dev/tty1crw--w----1 jenny tty 4, 1 Jan 23 11:41 /dev/tty1

joe:~> write jenny tty1Cau, pujdeme na obed?^C

Jenny na svém terminálu uvidí následující text:

Message from joe@lo.callhost.org on ptys/1 at 12:36 ...
Cau, pujdeme na obed?
EOF

Po přijetí zprávy může uživatel smazat obsah okna klávesami Ctrl+L. Příkazem mesg uživatel může zakázat příjem zpráv od jiných uživatelů (kromě správce systému). Pokud chcete zjistit, kteří přihlášení uživatelé mají povolen příjem zpráv, použijte příkaz who -w. Další podrobnosti nalez-nete na informačních stránkách příslušných příkazů.

Názvy skupin se mohou lišit

Volba názvů skupin je věcí konkrétní distribuce. Různé distribuce mohou používat různé názvy i různá jiná řešení.

Bootování, proces init, zastavení systému

Úvod

Jednou z největších výhod Linuxu je otevřený mechanismus spouštění a zastavení operačního systému, kdy se nahrávají zvolené programy se zvolenou konfigurací a uživatel tak má možnost libovolně nastavit celý proces spuštění systému i jeho korektní a řízené zastavení.
Kromě možnosti nastavit chování při spouštění a zastavení systému má otevřená povaha Linuxu další výhodu v tom, že je mnohem snazší přesně odhalit příčinu většiny problémů, k nimž může při spouštění a zastavování systému dojít. Základní pochopení celého procesu je pro každého uživatele linuxového systému velkou výhodou.

Velká část linuxových systémů používá k zavádění operačních systémů zavaděč lilo, LInux LOa-der. My budeme nicméně hovořit pouze o zavaděči GRUB, který je modernější, snáze se používá a nabízí více možností. Pokud byste potřebovali informace o zavaděči LILO, najdete je na manu-álových stránkách a v dokumentech HOWTO. Oba zavaděče podporují duální instalace; praktic-ké příklady a teoretické informace na toto téma naleznete v několika různých dokumentech HOWTO.

Bootovací proces

Při bootování systému x86 procesor spouští zaváděcí kód z programu BIOS (Basic Input/Output System), který je umístěn na konci operační paměti. Program BIOS je zapsán v permanentní pamě-ti a je vždy připraven k použití. BIOS poskytuje nejnižší úroveň rozhraní pro práci s periferními zařízeními a zajišťuje první kroky bootovacího procesu.

BIOS otestuje systém, vyhledá a zkontroluje periferie a následně vybere disk, z nějž se bude boo-tovat systém. Typicky hledá bootovatelné médium v disketové mechanice, CD-ROM mechanice a nakonec na pevném disku. Prohledávaná zařízení i jejich pořadí jde obvykle v BIOSu nastavit. Je-li na pevném disku nainstalován operační systém, BIOS nalezne záznam MBR (Master Boot Record), který je umístěn na prvním sektoru disku, načte jej do paměti a předá mu řízení.

MBR obsahuje instrukce, jak nahrát zavaděč GRUB (nebo LILO) s přednastaveným operačním systémem. MBR tedy nahraje zavaděč, který převezme řízení bootovacího procesu. U standardní instalace systému Red Hat Linux například GRUB načte z MBR nabídku bootovacích možností. Jak-mile má GRUB pohromadě všechny údaje potřebné k zavedení systému (ať už jako uživatelský vstup nebo načtením z konfiguračního souboru), najde zaváděcí soubor příslušného operačního systému a předá mu řízení.

Možnosti zavaděče GRUB

Popsaná metoda zavedení operačního systému se označuje jako *přímé zavedení*, protože zavaděč podle svého nastavení nahraje do paměti rovnou operační systém. Mezi kódem zavaděče a hlav-ním kódem operačního systému (například jádrem) už není žádná další softwarová vrstva. Způ-sob bootování jiných operačních systémů ovšem může být jiný. Například operační systémy MS DOS a MS Windows při své instalaci přepíší obsah MBR, aniž by se jakkoliv staraly o jeho stáva-jící obsah. Tím dojde ke zničení jakýchkoliv informací, které v MBR zapsaly jiné operační systé-my, například Linux. Operační systémy Microsoft, stejně jako různé jiné proprietární operační systémy, používají metodu *řetězeného zavedení*. V tomto případě MBR ukazuje na první sektor diskového oddílu s operačním systémem, kde se teprve nacházejí speciální soubory potřebné pro zavedení tohoto konkrétního systému.

GRUB podporuje obě dvě zaváděcí metody, takže jej lze použít pro spuštění téměř libovolného operačního systému, z většiny obvyklých souborových systémů a z téměř jakéhokoliv pevného disku, který BIOS rozezná.

Kromě toho nabízí GRUB ještě řadu dalších funkcí, mezi nejdůležitější patří:

GRUB obsahuje vlastní příkazové prostředí, jehož prostřednictvím je dosažena maximální flexibilita při zavádění operačního systému, umožňuje zjistit údaje o dostupných systémech a nastavit parametry jejich spuštění.
GRUB podporuje režim LBA (Logical Block Addressing), který je nutný pro práci s větši-nou IDE disků a se všemi SCSI disky. Dokud neexistoval režim LBA, byla oblast disku vyu-žitelná pro bootování omezena na 1 024 cylindrů, mimo tuto oblast BIOS nemohl z disku číst.
Konfigurace zavaděče se při každém spuštění systému načítá přímo z disku, není proto nutné modifikovat MBR pokaždé, když změníte nastavení zavaděče.

Podrobný popis chování zavaděče GRUB můžete získat příkazem info grub nebo přímo na webo-vých stránkách *http://www.gnu.org/software/grub.* Linuxový dokumentační projekt nabízí mimo jiné dokument HOWTO, popisující bootování více operačních systémů se zavaděčem GRUB (*http://www.tldp.org/HOWTO/Multiboot-with-GRUB.html*).

Proces init

Jakmile je jádro zavedeno, spustí program init z adresáře sbin. Po svém spuštění se proces init stává rodičem či prarodičem všech procesů, které se v linuxovém systému automaticky spustí. Jako první věc po svém spuštění init načte svůj konfigurační soubor /etc/inittab. Ten programu nařídí načtení inicializačních konfiguračních skriptů prostředí, které nastaví cesty, aktivují odkládací prostor a podobně. V zásadě se tyto skripty postarají o všechno, co je nutné v rámci inicializace systému provést – nastaví čas, sériové porty a tak dále. Dále soubor /etc/inittab říká, jak má být systém nastaven v jednotlivých úrovních běhu, a definu
je výchozí úroveň běhu. Úroveň běhu v zásadě představuje konfiguraci procesů. Všechny unixo-vé systémy mohou běžet s různými konfiguracemi procesů, například v jednouživatelském reži-mu, který se označuje jako úroveň běhu 1 či S. V tomto režimu se k systému může připojit pouze administrátor. Používá se ve speciálních případech údržby systému, kdy je nutno vyloučit poško-zení systémových či uživatelských dat. V tomto režimu jsou vypnuty všechny uživatelské služby. Jiná úroveň je například úroveň 6, která se používá pro restart systému. Předepsaným způsobem dojde k zastavení všech běžících služeb a systém se restartuje.

Obvyklé úrovně běhu jsou 3 pro textový režim a 5, která aktivuje grafické prostředí a přihlášení.
Další informace o úrovních běhu uvedeme v kapitole „Úrovně běhu“. Po zjištění výchozí úrovně běhu init načte a spustí všechny procesy, které mají na dané úrovni běžet – zjistí je z adresáře rc odpovídajícího zvolené úrovni. Nejprve init spustí všechny ukončo-vací skripty (jejich názvy začínají písmenem K) a následně všechny spouštěcí skripty (názvy zači-nají písmenem S) příslušné úrovně běhu, takže dojde ke korektnímu spuštění všech služeb a apli-kací. Tyto skripty můžete (samozřejmě jako *root*) spustit v kterémkoliv okamžiku i sami – napří-klad příkazem /etc/init.d/httpd stop nebo service httpd stop zastavíte webový server.

> Poznámka – Speciální případ
>
> Při spouštění systému se obvykle provedou skripty rc2.d a rc3.d. V tomto případě nedo chází k zastavení žádných služeb, definují se v nich pouze spouštěné služby.

Skripty, které fakticky spouštějí a zastavují jednotlivé služby, nejsou umístěny přímo v adresáři /etc/rc<x>.d. V tomto adresáři se nacházejí pouze symbolické odkazy na vlastní skripty uložené v /etc/init.d. Symbolický odkaz není nic jiného než soubor odkazující se na jiný soubor. Odkazy se používají z toho důvodu, že je lze na jednotlivých úrovních běhu vytvářet a mazat, aniž by to ovlivnilo vlastní skripty, které obsluhu služeb zajišťují.

Symbolické odkazy na jednotlivé skripty jsou očíslovány, takže je možné je spouštět v požadovaném pořadí. Pořadí, ve kterém jsou jednotlivé služby spouštěny či ukončovány, lze jednoduše změnit změnou názvu odkazu na skript, který zajišťuje obsluhu služby. Pokud má být jedna služ-ba spuštěna bezprostředně s jinou službou, mohou mít oba odkazy stejné číslo tak, jak to ukazu-je následující výpis adresáře /etc/rc5.d, kde se služby crond i xfs spouštějí odkazem, jehož název začíná znaky „S90“. V takovém případě budou skripty volány v abecedním pořadí.

```
[jean@blub /etc/rc5.d] ls
K15httpd@       K45named@       S08ipchains@    S25netfs@       S85gpm@
K16rarpd@       K46radvd@       S08iptables@    S26apmd@        S90crond@
K20nfs@         K61ldap@        S09isdn@        S28autofs@      S90xfs@
K20rstatd@      K65identd@      S10network@     S30nscd@        S95anacron@
K20rusersd@     K74ntpd@        S12syslog@      S55sshd@        S95atd@
K20rwalld@      K74ypserv@      S13portmap@     S56rawdevices@  S97rhnsd@
K20rwhod@       K74ypxfrd@      S14nfslock@     S56xinetd@      S99local@
K25squid@       K89bcm5820@     S17keytable@    S60lpd@
K34yppasswdd@   S05kudzu@       S20random@      S80sendmail@
```

Po spuštění služeb požadované úrovně běhu pokračuje skript /etc/inittab spuštěním procesu getty (textové přihlášení) pro všechny virtuální konzoly. Program getty otevře terminály, nastaví jejich režim a vypíše na nich výzvu k přihlášení. Po případném zadání jména a hesla spustí přihlašova-cí proces. Většina systémů otevírá šest virtuálních konzol, toto nastavení však lze v souboru init-tab změnit.

V souboru /etc/inittab se také nastavuje reakce systému na stisk kláves Ctrl+Alt+Del. Systém by nikdy neměl být restartován

natvrdo, proto se obvykle nastavuje, aby proces init na stisk uvede-ných kláves reagoval například spuštěním příkazu /sbin/shutdown -t3 -r now. Kromě toho lze v souboru /etc/inittab nastavit i reakce procesu init na výpadek napájení v případě, že je počítač napájen z UPS.

U většiny distribucí založených na RPM balíčcích se spouští grafické přihlášení v úrovni 5, kdy /etc/inittab zavolá skript /etc/X11/prefdm. Tento skript spustí zvoleného správce displeje podle nastavení v souboru /etc/sysconfig/desktop. Typicky je to gdm, pokud používáte Gnome, nebo kdm, používáte-li KDE. Oba správce však lze zaměnit nebo lze použít správce xdm, který je sou-částí standardní instalace X.

Jiné systémy mohou samozřejmě používat jiná řešení. Například v Debianu existuje pro každého správce displeje samostatný inicializační skript a o tom, který má být spuštěn, rozhoduje nastave-ní v souboru /etc/X11/default-display-manager. Další informace o grafickém prostředí naleznete v kapitole „Grafické prostředí“. Podrobnější údaje vám samozřejmě nabízí také dokumentace k vašemu systému.

V době bootování se dále načítá obsah adresářů /etc/default a/nebo /etc/sysconfig, v nichž se nastavuje celá řada parametrů a vlastností systému. V závislosti na používané distribuci může být výchozí nastavení systému uloženo i v jiných adresářích.

Kromě grafického prostředí se samozřejmě může spouštět i celá řada dalších služeb. Tak či onak, pokud vše proběhne správně, na konci bootovací procedury by se měla zobrazit výzva k přihlášení nebo rovnou přihlašovací obrazovka.

Jiné systémy

Popsali jsme si, jak funguje init verze System V na architektuře x86. Na jiných architektu-rách a v jiných distribucích se může způsob spouštění systému lišit. Některé systémy pou žívají init verze BSD, kde spouštěcí soubory nejsou rozděleny do samostatných adresářů /etc/rc<x>.d. Některé distribuce také místo adresáře /etc/init.d používají adresář /etc/rc.d/init.d.

Úrovně běhu

Základní myšlenka spouštění různých služeb na různých úrovních běhu vychází z toho, že různé systémy mohou být používány různými způsoby. Některé služby navíc nelze používat, pokud systém není v určitém stavu, například pokud neumožňuje přihlášení více uživatelů nebo pokud není aktivní síť.

Jsou situace, kdy záměrně potřebujete systém spustit v nižší úrovni. Příkladem může být oprava poškozených dat na disku v úrovni 1, kdy máte zaručeno, že v systému nebudou pracovat žádní uživatelé, anebo třeba server běžící v úrovni 3, bez grafického prostředí. V těchto případech nedá-vá smysl spouštět služby, které potřebují vyšší úroveň běhu, protože by stejně nepracovaly správ-ně. Tím, že je pro každou úroveň definováno, jaké služby se na ní mají spouštět, máte vždy zaru-čenu konzistenci procesu spouštění systému a snadno můžete přepnout systém z jednoho režimu do druhého bez nutnosti uvažovat, které služby je nutno zastavit a jaké jsou jejich závislosti.

Definované úrovně běhu jsou popsány v souboru /etc/inittab, příslušnou část zde uvádíme:

```
## inittab Tento soubor popisuje, jak má inicializační proces nastavit# systém na jednotlivých úrovních běhu.

# Možné úrovně běhu:# 0 - zastavení (NEPOUŽÍVAT jako výchozí!)# 1 - jednouživatelský režim# 2 - víceuživatelský režim bez NFS # (Nepoužíváte-li sí, stejné jako 3)# 3 - plný víceuživatelský režim# 4 - nepoužito# 5 - X11# 6 - reboot (NEPOUŽÍVAT jako výchozí!)# # Výchozí režim běhu:id:5:initdefault:
```

Úrovně 2 a 4 si můžete nastavit, jak uznáte za vhodné. Řada uživatelů dává přednost tomu, že úrovně 3 a 5 ponechá ve výchozím nastavení a úrovně 2 a 4 si nastaví přesně podle svých kon-krétních potřeb. Díky tomu se mohou snadno přepínat mezi specifickými konfiguracemi systému, aniž by narušili normální běh operací na standardních úrovních.

Pokud se počítač dostane do stavu, že kvůli porušenému souboru /etc/inittab nebude správně bootovat, nebo pokud se kvůli poškozenému souboru /etc/passwd nebudete moci přihlásit (či prostě jen zapomenete heslo), nabootujte do jednouživatelského režimu.

> Bez grafiky
>
> Pokud pracujete v textovém režimu, protože se neobjevila grafická přihlašovací obrazov-ka, můžete se zkusit přepnout na konzolu 7 nebo vyšší, kde se grafické prostředí normál-ně nachází. Pokud to nepomůže, zkuste příkazem who -r ověřit, jaká je aktuální úroveň běhu systému. Je-li jiná než 5, je možné, že systém standardně v grafickém režimu nena-bíhá. V takovém případě se obraťte na správce systému nebo si přečtěte man init. Změna režimu běhu se provádí příkazem init, přepnutím z textové do grafické konzoly a zpět se úroveň běhu nemění.

Snažili jsme se, aby byl popis úrovní běhu, skriptů a konfigurací co nejobecnější. Existuje samo-zřejmě celá řada variant, například Gentoo Linux ukládá skripty v adresáři /etc/runlevels. Jiné systémy nejprve projdou nižšími úrovněmi běhu a spustí všechny jejich skripty, až se propracují ke konečné požadované úrovni. Podrobnosti naleznete v dokumentaci ke svému systému.

Nástroje

Nástroje chkconfig nebo update-rc.d, máte-li je v systému nainstalovány, představují jednodu-ché rozhraní pro práci se soubory v adresářové hierarchii /etc/init.d. Tyto nástroje šetří admini-strátorovi práci s manipulací s řadou symbolických odkazů v adresářích /etc/rc[x].d.

Některé systémy nabízejí také nástroj ntsysv s textovým rozhraním, možná vám přijde jednoduš-ší než řádkové rozhraní příkazu chkconfig. V systému SuSE Linux najdete nástroje yast a ins-serv. Mandrake nabízí nástroj *DrakConf*, který mimo jiné umožňuje přepínat mezi úrovněmi 3 a 5.V Mandrivě byl přejmenován na Mandriva Linux Control Center.

Většina distribucí nabízí i grafické rozhraní pro konfiguraci služeb, podívejte se do dokumentace

ke svému systému.Všechny uvedené nástroje musíte spouštět jako *root*. Administrátor samozřejmě může spouštěnístužeb na jednotlivých úrovních zajistit i tak, že potřebné odkazy vytvoří v příslušných adresáříchručně.

Zastavení systému

Unixové systémy nejsou stavěny na to, aby je někdo vypínal, ale když už musíte, použijte příkaz

shutdown. S volbou -h vypne systém, s volbou -r jej restartuje.V některých systémech najdete příkazy reboot a halt, které zavolají shutdown a zajistí takkorektní restart nebo zastavení systému, raději si na ně ale nezvykejte, protože nemusí být k dis-pozici všude.

Jestliže se váš počítač po zastavení systému automaticky nevypne, musíte jej nakonec vypnoutručně. Nikdy jej však nevypínejte dříve, než se vypíše zpráva o tom, že byl systém skutečně zasta-ven a je možno jej vypnout. Netrpělivost by mohla mít za následek ztrátu dat.

Správa procesů

Pomáhejte správci systému

Správa systémových prostředků, tedy včetně procesů, je práce administrátora systému. Ani normálnímu uživateli však neuškodí, pokud se v této problematice orientuje, zejména pokud jde o optimální běh jeho vlastních procesů.

Budeme se zabývat teorií kolem výkonnosti systému, i když se nebudeme pouštět do témat, jako je hardwarová optimalizace a podobně. Soustředíme se místo toho na běžné problémy, na něž uživatel může narazit, a na způsoby, jak se může uživatel zasadit o optimální využití systémových prostředků. Jak dále uvidíme, ve většině případů se vyplatí napřed přemýšlet a až potom konat.

Jak dlouho to trvá?

Bash nabízí vestavěný příkaz time, jímž můžete změřit, jak dlouho trvalo provedení určitého pří-kazu. Měření je velmi přesné a lze je použít na jakýkoliv příkaz. Následující příklad ukazuje, že vygenerování této knihy trvalo zhruba minutu a půl:

billy:~/xml/src> time makeOutput written on abook.pdf (222 pages, 1619861 bytes).Transcript written on abook.log.

real 1m41.056suser 1m31.190ssys 0m1.880s

GNU příkaz time v adresáři /usr/bin (neplést si s vestavěným příkazem shellu) zobrazuje více informací, které lze navíc různě formátovat. Mimo jiné zobrazí návratový kód programu a celko-vý zabraný čas. Zkusíme změřit stejný příkaz jako v předchozím příkladu:

billy:~/xml/src> /usr/bin/time makeOutput written on abook.pdf (222 pages, 1595027 bytes).Transcript written on abook.log.

Command exited with non-zero status 2
88.87user 1.74system 1:36.21elapsed 94%CPU

(0avgtext+0avgdata 0maxresident)k 0inputs+0outputs (2192major

+30002minor)pagefaults 0swaps

Jako vždy naleznete podrobnosti na informačních stránkách.

Výkon

Pro uživatele znamená slovo „výkon“ rychlou odezvu na příkazy. Pro správce systému to ale zna-mená mnohem více, protože správce musí optimalizovat výkon systému jako celku, tedy pro uži-vatele, všechny programy a démony. Výkon systému může záviset na tisícovce maličkostí, které příkaz time nezjistí:

program je špatně naprogramovaný nebo nevyužívá počítač správně,
přístup k diskům, řadičům, displeji, různým rozhraním a podobně,
dostupnost vzdálených systémů (výkon sítě),
počet uživatelů, počet skutečně současně pracujících uživatelů,
denní doba
a další...

Zatížení systému

Stručně řečeno, zatížení závisí na tom, co je pro daný systém obvyklé. Na starém P133 mi běží firewall, SSH server, souborový server, směrovací démon, poštovní server, proxy server a několik dalších služeb. Počítač obsluhuje sedm uživatelů a průměrné zatížení je trvale 0. Viděl jsem už (víceprocesorové) systémy, které se chovaly bezproblémově při zatížení 67. Existuje jediná meto-da, jak zjistit, co je normální – pravidelně zatížení systému sledujte. Pokud tuto hodnotu sledovat nebudete, můžete okamžitý stav systému pouze odhadnout na základě reakční doby příkazové řádky, což je velmi hrubé měření, jelikož tato doba je ovlivněna stovkou různých faktorů.

Nezapomínejte, že různé systémy se budou při stejném zatížení chovat různě. Například systém s grafickou kartou s podporou hardwarové akcelerace nebude mít problém s renderováním 3D obrázků, stejný systém s levnou VGA kartou se při stejné operaci výrazně zpomalí. Moje stará P133 by měla problémy při spuštění X serveru, na moderních systémech se to na zátěži téměř neprojeví.

Můžu jako uživatel něco udělat?

Může vás zpomalit nevhodně nastavené prostředí. Pokud máte nastaveno hodně proměnných pro-středí (místo proměnných shellu) a dlouhé a neoptimalizované vyhledávací cesty (chyby v nasta-vení proměnné PATH), může nalezení a čtení dat trvat systému déle.

Používáte-li X, mohou procesor hodně zatěžovat správci oken a pracovní plochy. Velmi efektní pracovní plochy něco stojí, i když je máte zadarmo – počet různých doplňků a aplikací může být obrovský. Základem je vždy střídmost, pokud tedy nechcete každý rok kupovat nový počítač.

Priorita

Priorita či význam úlohy je definován hodnotou *nice*. Program s vyšší hodnotou *nice* je k ostat-ním uživatelům, programům a systému „přátelštější“, nejde o důležitou úlohu. Čím je hodnota *nice* nižší, tím je úloha významnější a tím větší má nárok na systémové prostředky.

Snižovat nároky úlohy zvýšením její hodnoty *nice* má význam pouze u procesů, které mají velkou spotřebu procesorového času (překladače, matematické programy a podobně). Procesy, které používají velké množství vstupně-výstupních operací, jsou řízeny systémem a jejich priorita je automaticky vyšší – například vstup z klávesnice má vždy nejvyšší prioritu.

Nastavení hodnoty *nice* se provádí příkazem nice. Většina systémů navíc nabízí i příkaz renice, který umožňuje změnit hodnotu *nice* již spuštěného programu. Podrobnosti naleznete jako obvy-kle v manuálových stránkách.

> Interaktivní programy
>
> V žádném případě NENÍ rozumné manipulovat s hodnotou *nice* interaktivních programů a úloh běžících na popředí.

Většinou tyto příkazy používá správce systému. Na manuálových stránkách se dozvíte další podrobnosti o speciálních funkcích těchto příkazů, které jsou dostupné pouze administrátorovi.

Procesorový čas

Na každém linuxovém systému chce procesorový čas současně používat mnoho programů, dokonce i když jste jediným uživatelem systému. Každý program potřebuje ke svému proběhnu-tí určitý počet procesorových cyklů. Mohou nastávat situace, kdy volný procesorový čas není k dispozici, protože je procesor příliš zatížen. Příkaz uptime je velmi nepřesný, protože zobrazu-je pouze průměrné hodnoty zatížení (a pro srovnání tedy potřebujete znát normální stav), i tak ale může být užitečný. Pokud máte podezření, že pomalé reakce systému jsou způsobeny vysokým zatížením procesoru, můžete provést následující akce:

Náročné programy spouštět v době, kdy je zatížení systému nízké (například v noci). Více v následující části o plánování.
Nenechávejte systém dělat zbytečnou práci – zastavte démony a programy, které nepouží-váte, místo náročného find používejte locate...
Velké úlohy spouštějte s nízkou prioritou.

Pokud v konkrétní situaci nemůžete žádné z těchto řešení použít, možná budete muset upgradovat procesor.

Paměové prostředky

Pokud právě spuštěná úloha potřebuje více paměti, než je fyzicky k dispozici, systém se nezhroutí, ale začne stránkovat či swapovat – začne se využívat odkládací prostor na disku a bude docházet k přesunům mezi fyzickou pamětí a diskem, kam se budou odkládat části programů, nebo dokon-ce celé programy, aby se uvolnila potřebná část fyzické paměti. Systém se tím výrazně zpomalí, protože přístup na disk je mnohem pomalejší než přístup do fyzické paměti. Využití paměti a odkládacího prostoru můžete zjistit příkazem top, systémy s knihovnou glibc nabízejí vizualiza-ci použití paměti pomocí příkazů memusage a memusagestat.

Pokud zjistíte, že se používá hodně paměti a odkládacího prostoru, můžete vyzkoušet:

Zabít, ukončit nebo snížit prioritu programů s velkou spotřebou paměti.
Přidat paměť (a případně zvětšit velikost odkládacího prostoru).
Optimalizovat výkon systému, což je činnost mimo rámec této příručky. Návody naleznete v literatuře uvedené v kapitole „Kam dál“.

Vstupně-výstupní prostředky

Přestože vstupně-výstupní operace představují významný faktor ovlivňující chování systému, k jejich měření nejsou k dispozici dobré nástroje. Z příkazů ps, vmstat a top můžete získat před-stavu o tom, kolik programů čeká na vstupně-výstupní operace. Příkazem netstat zjistíte statisti-ky síťových rozhraní, příkazem iostat získáte stručný přehled obecného využití vstupně-výstup-ních rozhraní, neexistuje však žádný nástroj, který by měřil reakční statistiky vstupně-výstupních rozhraní jako celku. Existují různé grafické programy, které vizualizují výstupy právě uvedených příkazů do snáze čitelné a názornější podoby.
Každé zařízení má své specifické problémy, obecně ale platí, že úzkým místem vstupně-výstup
ních operací jsou propustnost síťových rozhraní a propustnost disků.
Síťové problémy:

Přetížení sítě: Objem dat přepravovaných po síti je větší než přenosová kapacita sítě, což vede ke zpomalení všech úloh souvisejících se sítí u všech uživatelů. Problém lze někdy řešit pročištěním sítě (vypnutím nepotřebných protokolů a služeb) nebo rekonfigurací sítě (rozdělením na podsítě, výměnou opakovačů za přepínače, instalací rychlejších rozhraní).
Problémy s integritou sítě: Dochází k nim při chybách přenosu dat v síti. Řešením je pouze nalezení a výměna chybujícího prvku.

Diskové problémy:

Nízká přenosová rychlost jednotlivého procesu – nedostatečná rychlost čtení či zápisu pro daný proces.
Nízká agregovaná přenosová rychlost – celková disková prostupnost nepostačuje všem potřebným programům.

Tyto problémy se obtížně detekují, a pokud je příčina v hardwaru, k jejich řešení je obvykle zapo-třebí změna hardwarové konfigurace tak, aby se rozdělily datové toky mezi sběrnicemi, řadiči a disky. Jedna možnost je použít RAID s konfigurací optimalizovanou na propustnost vstupně -výstupních operací. Druhou možností je upgrade na rychlejší sběrnice, řadiče či disky.

Jestliže problém není způsobem přetížením, může se jednat o hardwarovou závadu nebo špatné připojení k systému. Pro začátek můžete zkontrolovat kontakty, konektory a zásuvky.

Uživatelé

Podle využití systémových prostředků lze uživatele rozdělit do několika skupin:

Uživatelé, kteří spouštějí hodně malých úloh – typicky začínající uživatelé.
Uživatelé, kteří spouštějí relativně málo náročných úloh – typicky uživatelé, kteří spou-štějí simulační, výpočetní nebo emulační programy s velkými nároky na paměť. Tito uži-vatelé obvykle rovněž manipulují s velkými datovými soubory.
Uživatelé, kteří spouštějí málo úloh s velkou spotřebou procesorového času – typicky vývojáři.

Jak vidíte, požadavky jednotlivých kategorií uživatelů se mohou lišit a může být obtížné uspoko-jit každého. Pokud spravujete víceuživatelský systém, je rozumné (a zábavné) vysledovat zvyky jednotlivých uživatelů a systému, abyste pak mohli co nejlépe vyhovět konkrétním potřebám.

Grafické nástroje

Pro grafická prostředí je k dispozici celá řada monitorovacích nástrojů. Následující obrázek uka-zuje Gnome System Monitor, který mimo jiné dokáže zobrazit a hledat informace o procesech a umí sledovat využití systémových prostředků.

Existuje také celá řada appletů, které můžete nainstalovat na panel nástrojů a které sledují napří-klad využití disku, paměti nebo procesoru. Další malá aplikace pro sledování zátěže systému je například xload. Svůj oblíbený program si budete muset najít sami.

Přerušení procesů

Jako neprivilegovaný uživatel můžete ovlivnit pouze své vlastní procesy. Už jsme viděli, jak můžeme vypsat běžící procesy a

filtrovat je podle majitele, včetně případných omezení, která se mohou projevit. Pokud zjistíte, že některý z vašich procesů spotřebovává příliš mnoho systémových prostředků, můžete udělat dvě věci:

Přimět proces ke snížení spotřeby prostředků bez jeho přerušení.
Úplně proces ukončit.

Pokud proces nechcete přerušit, ale chcete omezit jeho nároky, můžete změnit jeho prioritu pří-kazem renice. Kromě příkazů nice a renice je dalším užitečným nástrojem pro nalezení problé-mových procesů a snížení jejich priority příkaz top.
Všimněte si hodnoty ve sloupci „NI“, s největší pravděpodobností to bude záporné číslo. Zmáčk-něte r a zadejte PID procesu, jehož prioritu chcete změnit. Pak zadejte novou hodnotu, například „20“. Toto nastavení znamená, že od této chvíle bude proces spotřebovávat maximálně jednu pěti-nu procesorového času.

Typickým příkladem procesů, které budete chtít nechat běžet byť se sníženou prioritou, jsou emu
látory, virtuální stroje, překladače a podobně. Pokud budete chtít proces ukončit, protože „zatuhl“ nebo provádí enormní množství vstupně -výstupních operací či spotřebovává značnou část jiných systémových prostředků, použijte příkaz kill. Pokud to půjde, zkuste nejprve proces ukončit „měkce“, signálem *SIGTERM*. Tímto signálem procesu říkáte, aby korektně ukončil svou činnost tak, jak to má naprogramováno:

joe:~> ps -ef | grep *mozilla* joe 25822 1 0 Mar11 ? 00:34:04 /usr/lib/mozilla-1.4.1/mozilla-

joe:~> kill -15 *25822*

Výše uvedený příklad ukazuje, jak uživatel *joe* ukončil prohlížeč Mozilla.V některých případech nemusí být tak snadné se procesu zbavit. Máte-li na to čas, zkuste nejpr-ve proces přerušit signálem *SIGINT*. Pokud by to nestačilo, použijte nejsilnější signál, *SIGKILL*.Následující příklad ukazuje ukončení úplně zamrzlého prohlížeče:

joe:~> ps -ef | grep *mozilla* joe 25915 1 0 Mar11 ? 00:15:06 /usr/lib/mozilla-1.4.1/mozilla-

joe:~> kill -9 *25915*

joe:~> ps -ef | grep *25915* joe 2634 32273 0 18:09 pts/4 00:00:00 grep 25915
V těchto případech je rozumné ještě jednou vypsat seznam procesů a vyfiltrovat příslušné PID a ujistit se tak, že proces skutečně skončil. Jestliže se ve výpisu objeví pouze samotný příkaz grep, máte jistotu, že se vám podařilo proces zastavit.

Jedním z procesů, jejichž zabití je obtížné, je samotný shell. To je samozřejmě dobře, kdyby jej bylo tak snadné zabít, ukončil by se pokaždé, když na příkazovém řádku omylem zmáčknete Ctrl+C, protože tato kombinace kláves je ekvivalentem zaslání signálu *SIGINT*.

> UNIX bez rour je nepředstavitelný
>
> O použití rour (|) k přesměrování výstupu jednoho příkazu na vstup dalšího příkazu bude
> me hovořit v následující kapitole.

V grafickém prostředí můžete použít jednoduchý program xkill. Zadejte tento příkaz, zmáčkněte Enter a zvolte okno aplikace, kterou chcete zavřít. Tento program ovšem standardně posílá sig-nál *SIGKILL*, takže jej používejte pouze tehdy, pokud aplikace úplně zatuhne.

Plánování procesů

Využijte nevyužívaný čas!

Linuxový systém může být vytížen spoustou věcí, obvykle je však vytížen pouze v pracovní době. Ať už se bavíme o kancelářském počítači, serveru nebo domácí stanici, většina systémů se ráno, večer, v noci a o víkendech nudí. Využití tohoto volného času je mnohem levnější než nákup tak výkonného počítače, aby dokázal uspokojit všechny vaše požadavky současně.

Existují tři metody odloženého spuštění programu:

Pokud chcete program jen na chvíli pozdržet, použijte příkaz sleep. Běh programu bude pokračovat po uplynutí nastaveného intervalu.
Pokud chcete program spustit v určitý čas, použijte příkaz at. Úloha bude spuštěna v oka-mžiku, který nadefinujete.
Pokud chcete nějaký program spouštět pravidelně každý měsíc, týden, den či hodinu, pou-žijte příkaz cron.

Nyní si jednotlivé možnosti podrobněji popíšeme.

Příkaz sleep

Informační stránka příkazu sleep je pravděpodobně jedna z nejkratších vůbec. Příkaz nedělá nic jiného, než že čeká. Standardně čeká zadaný počet sekund.
K čemu je to dobré? Několik praktických příkladů:Kolega vám zavolá, že jdete za půl hodiny na oběd. Vy jste ale docela zabraní do práce a klidněby se mohlo stát, že na oběd zapomenete. Tak rychle zadáte:

```
(sleep 1800; echo "Oběd!")&
```

Jiná situace. Je pět hodin, chcete jít domů, potřebujete ale spustit ještě nějaký program, a systém je momentálně příliš zatížen jinými úlohami. Pokud z nějakého důvodu nechcete nebo nemůžete použít příkaz at, můžete zadat například:

```
(sleep 10000; program)&
```

Aby takový postup fungoval, nesmí být v systému nastaveno automatické odhlašování uživatelů a musíte se korektně odhlásit z terminálu nebo zůstat přihlášeni a terminál zamknout. Můžete také použít příkaz screen.

Pokud byste například potřebovali vytisknout několik velkých souborů, chcete ale dát ostatním uživatelům šanci mezi tím také něco vytisknout, můžete zadat:

```
lp dlouhý_soubor; sleep 900; lp další_dlouhý_soubor; sleep 900; lp ještě_jeden
```

O tisku souborů budeme podrobněji hovořit v kapitole „Tiskárny a tisk“.Programátoři příkaz sleep často používají k pozastavení běhu programu či skriptu na určitou dobu.

Příkaz at

Příkaz at naplánuje spuštění příkazu v určitou dobu. Pokud nestanovíte jinak (viz manuálovéstránky), příkaz se spustí ve vašem výchozím shellu.Volby příkazu at jsou uživatelsky velmi příjemné, jak ukazují následující příklady:

```
steven@home:~> at tomorrow + 2 days warning: commands will be executed using (in order) a) $SHELL
        b) login shell c) /bin/sh at> cat reports | mail myboss@mycompany at> <EOT> job 1 at 2001-06-16 12:36
```

Zadávání příkazů k odloženému spuštění ukončíte stiskem Ctrl+D.V tomto příkladu uživatel *steve* zkombinoval dva příkazy dohromady – o těchto praktikách bude-me hovořit v kapitole „Přesměrování vstupu a výstupu“.

```
steven@home:~> at 02:37 warning: commands will be executed using (in order) a) $SHELL
        b) login shell c) /bin/sh at> cd new-programs at> ./configure; make at> <EOT> job 2 at 2001-06-14 02:00
```

Přepínačem -m zadáte, aby byl uživateli zaslán e-mail po provedení úlohy, případně pokud se úlohu z nějakého důvodu provést nepodařilo. Příkazem atq vypíšete úlohy čekající ve frontě. Před zadáním vlastních úloh se vždy podívejte, co už ve frontě čeká, abyste náhodou nenaplánovali spuštění vaší úlohy současně s jinou úlohou. Pokud si to později rozmyslíte, můžete příkazem atrm úlohu z fronty odstranit.

Je rozumné volit neobvyklé časy spuštění úloh, protože jak uvidíme v následující části, v „kula-tých časech“ se obvykle spouštějí systémové úlohy. Například přesně v jednu hodinu v noci se spouštějí indexace souborové databáze, takže naplánovat spuštění jiné úlohy na tuto dobu není moc rozumné. Pokud chcete zabránit současnému běhu mnoha úloh, můžete použít také příkaz batch, který řadí procesy do fronty a předává je systému ke zpracování jeden po druhém, takže se předejde špičkám v zatížení systému. Více informací naleznete na informačních stránkách.

Cron a crontab

Pravidelné spouštění programů zajišťuje démon *cron*. Ze záznamů v systémové a v uživatelských tabulkách („crontabech“) zjistí, které programy a kdy mají být spuštěny. Systémový crontab může měnit pouze uživatel root, ostatní uživatelé mohou měnit jen své crontaby. Na některých systé-mech nemusí mít někteří uživatelé tuto možnost vůbec.

Při svém spuštění *cron* hledá v adresáři /var/spool/cron/ soubory pojmenované stejně jako uži-vatelské účty v souboru /etc/passwd, dále prohledává adresář /etc/cron.d/ a načítá soubor /etc/crontab. Každou minutu pak všechny takto získané údaje kontroluje, jestli nemá být právě něco spuštěno. Příslušné programy pak spouští jako uživatel, který vlastní příslušný crontab, a tomuto uživateli následně e-mailem pošle eventuální výstup programů.

Na systémech, které používají Vixie verzi démona *crond*, jsou úlohy spouštěné každou hodinu, den, týden a měsíc kvůli přehlednosti zapsány v samostatných podadresářích adresáře /etc, zatím-co standardní unixová verze démona pracuje pouze s jedním velkým souborem.

Příklad crontabu Vixie verze démona může vypadat takto:

```
[root@blob /etc]# more crontab SHELL=/bin/bash PATH=/sbin:/bin:/usr/sbin:/usr/bin MAILTO=root HOME=/

# run-parts # commands to execute every hour 01 * * * * root run-parts /etc/cron.hourly # commands to execute every day 02 4 * * * root run-parts /etc/cron.daily # commands to execute every week 22 4 * * 0 root run-parts /etc/cron.weekly commands to execute every month 42 4 1 * * root run-parts /etc/cron.monthly
```

Nejprve se nastaví potřebné proměnné a pak už následuje samotné plánování. Na každém řádku je definována jedna úloha, řádek vždy začíná pěticí časových údajů. První údaj definuje minutu (0 až 59), druhý údaj hodinu (0 až 23), třetí údaj den v měsíci (1 až 31), čtvrtý údaj měsíc (1 až 12) a pátý den v týdnu (0 až 7, hodnoty 0 i 7 znamenají neděli). Hvězdička na daném místě zname-ná „pokaždé“. Je možno použít i seznamy hodnot, takže spouštění každé pondělí až pátek nasta-víte tak, že v posledním poli uvedete „1–5“, chcete-li úlohu spouštět v pondělí, středu a pátek, uvedete „1,3,5“.

Dalším údajem je uživatel, pod jehož účtem se má program spustit. Zbytek řádku už je samotný název spouštěného programu a jeho parametry. Výše uvedený příklad pochází z konfigurace Vixie varianty démona *cron*, kde v nastavených intervalech spouští uživatel *root* program run-parts, jehož parametrem je název adresáře. V příslušných adresářích jsou jako skripty definovány vlast-ní úlohy, které mají být v daném čase spuštěny. Například následující krátký skript zajišťuje aktu-alizace databáze používané příkazem locate:

```
billy@ahost cron.daily]$ cat slocate.cron #!/bin/sh renice +19 -p $$ >/dev/null 2>&1 /usr/bin/updatedb -f "nfs,smbfs,ncpfs,proc,devpts" -e \
"/tmp,/var/tmp, /usr/tmp,/afs,/net"
```

Bezpečný způsob, kterým uživatelé mohou modifikovat svůj crontab, je příkaz crontab -e. Zabrá-ní tomu, aby uživatel omylem současně neotevřel více kopií crontabu. Výchozím editorem je vi(viz kapitolu „Textové editory“); pokud vám ale více vyhovuje grafické prostředí, můžete použítjakýkoliv jiný editor, například gvim nebo gedit.
Po skončení editace vám systém oznámí, že došlo k instalaci nového crontabu.Následující záznam v crontabu uživateli *billy* každou středu odpoledne připomene, že má jít cvi-čit:

```
billy:~> crontab -l# DO NOT EDIT THIS FILE - edit the master and reinstall.# (/tmp/crontab.20264 installed on Sun Jul 20 22:35:14 2003)#
(Cron version -- $Id: chap4.xml,v 1.23 2006/01/07 13:47:14 tille Exp $)38 16 * * 3 mail -s "sports evening" billy
```

Po uložení změn vám systém oznámí, že došlo k instalaci nového crontabu. Aby se změny proje-vily, není nutné démona crond restartovat. Následující příklad ukazuje, jak *billy* přidal záznam pro spouštění zálohovacího skriptu:

```
billy:~> crontab -e 38 16 * * 3 mail -s "sports evening" billy 4 4 * * 4,7 /home/billy/bin/backup.sh

<-- zapsat a ukončit-->

crontab: installing new crontab

billy:~>
```

Skript backup.sh se spouští každou středu a neděli. O vytváření skriptů budeme podrobněji hovořit v kapitole „Shellové skripty“, případně v jiné části knihy (Bash pro začátečníky). Nezapo-mínejte, že případný výstup příkazů se posílá e-mailem majiteli příslušného crontabu. Pokud v systému není nastavena poštovní služba, možná výstupy najdete v lokální schránce, tedy v tex-tovém souboru /var/spool/mail/<uživatelské_jméno>.

Kdo spouští příkazy

Nemusíte nijak specifikovat, pod jakým uživatelským účtem mají být příkazy spuštěny.
Automaticky se spouštějí s právy uživatele, který vlastní příslušný crontab.

Shrnutí

Linux je víceuživatelský víceúlohový operační systém, který používá mechanismus obsluhy procesů analogický k Unixu. Rychlost provádění příkazů může záviset na tisícovce maličkostí. Mimo jiné jsme se seznámili s celou řadou nových příkazů pro zobrazení procesů a manipulaci s nimi. Toto je jejich seznam:

Příkaz	Význam
	Naplánuje úlohu k pozdějšímu spuštění.
	Vypíše seznam naplánovaných úloh.
	Smaže úlohu se zadaným číslem.
	Spustí příkaz v okamžiku, kdy to dovolí zatížení systému.

Příkaz	Význam
	Edituje crontab spouštějícího uživatele.
	Zastaví systém.

Nastaví zadanou úroveň běhu.

Vypíše právě probíhající úlohy.

Ukončí proces.

Řídí práva zápisu k vašemu terminálu.

Vypisuje síová spojení, směrovací tabulky, statistiky rozhraní, maškarádovaná spojení a člen
ství v multicastových skupinách.

Spustí program se změněnou prioritou plánování.

Vypíše běžící procesy.

Zobrazí strom procesů.

Restartuje systém.

Mění prioritu běžícího procesu.

Zastaví systém.

Čeká zadanou dobu.

Změří spotřebu času nebo hlásí využití prostředků.

Vypíše nejnáročnější procesy.

Ukáže, jak dlouho je systém spuštěn.

Zobrazí statistiky virtuální paměti.

Ukáže, kdo je přihlášen a co dělá.

Pošle zprávu na všechny terminály.

Ukáže, kdo je přihlášen.

Pošle zprávu jinému uživateli.

Příkazy pro práci s procesy

Cvičení

Následující cvičení vám umožní lépe se seznámit s procesy běžícími v systému.

Obecné

Spusťte v jednom terminálu příkaz top a další cvičení provádějte v jiném.
Spusťte příkaz ps.
V manuálových stránkách si zjistěte, jak vypíšete všechny své procesy.
Spusťte příkaz find /. Jaký má vliv na zatížení systému? Přerušte příkaz.
V grafickém režimu spusťte na popředí program xclock. Přepněte jej na pozadí. Ukončete-te jej příkazem kill.
Spusťte příkaz xcalc rovnou na pozadí, aby zůstal přístupný prompt terminálu.
Co udělá příkaz kill -9 -1?
Otevřete dva terminály nebo terminálová okna a příkazem write pošlete zprávu z jedno-ho do druhého.
Spusťte příkaz dmesg. Co říká?
Jak dlouho trvá proběhnutí příkazu ls v aktuálním adresáři?
Jak na základě položek v adresáři /proc vlastněných vaším UID zjistíte, kterým procesům tyto položky odpovídají?
Jak dlouho systém běží?
Který terminál právě používáte?
Vyjmenujte tři procesy, které nemohou mít jako původního rodiče proces init.
Vyjmenujte tři příkazy, které používají SUID režim, a vysvětlete proč.
Vyjmenujte příkazy, které na vašem systému obecně způsobují nejvyšší zátěž.

Bootování, init a podobně

Můžete systém restartovat jako normální uživatel? Proč?
Podle aktuální úrovně běhu uveďte, jaké kroky proběhnou při zastavování systému.
Jak změníte úroveň běhu? Přepněte se z úrovně 3 do 5 a naopak.
Zjistěte seznam všech služeb a démonů, které se spouštějí při startu systému.
Jaké jádro se zavádí při spouštění systému?
Předpokládejme, že při spouštění systému musíte spouštět nějakou exotickou službu. Doposud se po spuštění systému musíte přihlásit a spustíte ji skriptem deliver_pizza ve svém domovském adresáři. Co musíte udělat, aby se daná služba spouštěla

automaticky v úrovni běhu 4?

Plánování procesů

Použijte příkaz sleep k připomenutí, že za deset minut budete mít ohřátou večeři.
Vytvoře úlohu naplánovanou příkazem at, která za půl hodiny zkopíruje všechny soubory z vašeho domovského adresáře do /var/tmp. Vytvořte si pro tento účel ve /var/tmp pod-adresář.
Naplánujte provedení této úlohy každé poledne ve všední dny.
Ověřte, že to funguje.
Vytvořte chybnou položku crontabu, zadejte například neexistující příkaz coppy namísto cp. Co se stane, až bude položka spuštěna?

Přesměrování vstupu a výstupu

V této kapitole se blíže seznámíme s mocným unixovým mechanismem přesměrování vstupu, výstupu a chybového výstupu. Budeme hovořit o následujících tématech:

Standardní vstup, výstup a chybový výstup
Operátory přesměrování
Jak použít výstup jednoho příkazu jako vstup pro další příkaz
Jak uložit výstup příkazu do souboru pro pozdější použití
Jak do jednoho souboru zapsat výstup více příkazů
Přesměrování vstupu
Obsluha standardních chybových hlášení
Kombinace přesměrování vstupu, výstupu a chybového výstupu
Výstupní filtry

Jednoduché přesměrování

Co je to standardní vstup a výstup

Většina linuxových příkazů načte nějaký vstup, například soubor nebo parametry příkazu, a vytvo-ří výstup. Standardně se vstup zadává na klávesnici a výstup se zobrazuje na obrazovce. Kláves-nice představuje standardní vstupní zařízení (*stdin*), obrazovka nebo příslušné terminálové okno pak standardní výstupní zařízení (*stdout*).

Linux je ovšem pružný systém a toto výchozí chování je možné modifikovat. Jako standardní výstup přísně sledovaného serveru může například sloužit tiskárna.

Operátory přesměrování Přesměrování výstupu operátory >

a |

Občas můžete potřebovat, aby se výstup nějakého příkazu zapsal do souboru, nebo budete chtít výstup jednoho příkazu nechat zpracovat jiným příkazem. Takové činnosti se říká přesměrování výstupu. Přesměrování se provede buď operátorem „>“ (větší než) nebo operátorem „|“ (roura), který pošle standardní výstup jednoho příkazu na standardní vstup dalšího příkazu.
Jak už jsme viděli, příkaz cat vypíše obsah souboru či souborů na standardní výstup. Přesměru-jeme-li jej do nějakého souboru, dojde k vytvoření nového souboru – případně k přepsání stáva-jícího, pokud už stejnojmenný soubor existuje, takže buďte opatrní.

```
nancy:~> cat test1 nějaký text

nancy:~> cat test2 nějaký jiný text

nancy:~> cat test1 test2 > test3

nancy:~> cat test3 nějaký text nějaký jiný text
```

Pozor na přepsání!

Při přesměrování výstupu si dávejte pozor, abyste si nepřepsali již existující (důležitý) sou-bor. Řada shellů včetně bashe obsahuje vestavěnou funkci, která vás před takovým omy-lem ochrání – noclobber. Podrobnosti se dozvíte na informační stránce. V bashi zabrání-te neúmyslnému přepsání souborů tak, že do konfiguračního souboru .bashrc přidáte pří-kaz set -o *noclobber*.

Přesměrujete-li do existujícího souboru „nic“, dojde k vyprázdnění souboru:

```
nancy:~> ls -l list -rw-rw-r--1 nancy nancy 117 Apr 2 18:09 list

nancy:~> > list

nancy:~> ls -l list -rw-rw-r--1 nancy nancy 0 Apr 4 12:01 list
```

Tato operace se označuje *truncating*.Stejný způsob přesměrování do neexistujícího souboru vytvoří prázdný soubor se zadanýmnázvem:

```
nancy:~> ls -l newlist ls: newlist: No such file or directory

nancy:~> > newlist

nancy:~> ls -l newlist -rw-rw-r--1 nancy nancy 0 Apr 4 12:05 newlist
```

Další příklady těchto způsobů přesměrování si ukážeme v kapitole „Home, sweet /home“.Nyní několik příkladů na použití roury:Chceme prohledat text, najít v něm všechny řádky obsahující „nějaký_text“ a vyloučit z nich ty,

které obsahují „jiný_text“:

grep *nějaký_text* soubor | grep -v *jiný_text*
Chceme vypsat obsah adresáře po stránkách:

ls -la | less

Hledáme soubory v adresáři:

ls -l | grep *část_názvu_souboru*

Přesměrování vstupu

V jiných případech můžete potřebovat předat soubor jako vstup příkazu, který normálně se sou-bory nepracuje. Takovéto přesměrování vstupu se provádí operátorem „<“ (menší než). Následující příklad ukazuje, jak někomu poslat soubor:

```
andy:~> mail mike@somewhere.org < to_do
```

Pokud uživatel *mike* existuje jako lokální uživatel systému, nemusíte zadávat celou e-mailovou adresu. Pokud ovšem posíláte e-mail mimo lokální systém, je úplná adresa nutná. Uvedené řešení sice není na první pohled tak zjevné jako začátečnický ekvivalent cat soubor | mail *adresa*, je to ovšem metoda daleko elegantnější a efektivnější, protože ušetří jedno vytvo-ření nového procesu.

Kombinované přesměrování

Následující příklad kombinuje přesměrování vstupu i výstupu. Kontrolují se překlepy v souboru text.txt a výsledek se zapisuje do souboru error.log:

spell < text.txt > error.log

Následujícím příkazem vypíšeme všechny možnosti, jak v příkazu less prohlížet („prozkoumat“) více souborů:

```
mike:~> less --help | grep -i examine
   :e [file] Examine a new file.
   :n * Examine the (N-th) next file from the command line.
   :p * Examine the (N-th) previous file from the command line.
   :x * Examine the first (or N-th) file from the command line.
```

Všimněte si volby -i příkazu grep, ta zajistí, že se při prohledávání nezohledňuje velikost písmen. Budete-li si chtít výpis uložit pro pozdější použití, můžete jej přesměrovat do souboru:

```
mike:~> less --help | grep -i examine > examine-files-in-less
```

```
mike:~> cat examine-files-in-less
   :e [file] Examine a new file.
   :n * Examine the (N-th) next file from the command line.
   :p * Examine the (N-th) previous file from the command line.
   :x * Examine the first (or N-th) file from the command line.
```

Přesměrování výstupu jednoho příkazu na vstup dalšího příkazu je možné neomezeně řetězit, jedi-nou podmínkou je, aby příkazy v řetězci přijímaly vstup ze standardního vstupu a vypisovaly výstup na standardní výstup. Některé příkazy to samy o sobě nedělají, ale lze jim to vhodnou vol-bou nařídit. Pokud tedy narazíte na nějakou neočekávanou chybu, přečtěte si dokumentaci (tedy manuálové a informační stránky) k používaným příkazům.

Znovu upozorňujeme, abyste omylem nepřesměrovali výstup do již existující důležitého souboru. Přesměrováním dojde k přepsání původního obsahu souboru.

Operátor >>

Pokud pro přesměrování výstupu použijete dvojici znaků „větší než“, nedojde k přepsání existují
cího souboru, výstup se přidá na jeho konec.
Například:

```
mike:~> cat wishlist víc peněz míň práce

mike:~> date >> wishlist

mike:~> cat wishlist víc peněz míň práce Thu Feb 27 20:23:07 CET 2007
```

Příkaz date by za normálních okolností vypsal datum a čas na obrazovku, v uvedeném příkladu je ovšem připsal do souboru wishlist.

Pokročilejší možnosti přesměrování

Použití souborových deskriptorů

Existují tři různé vstupy a výstupy, každý z nich má svůj identifikátor, takzvaný deskriptor souboru:

standardní vstup: 0,
standardní výstup: 1,
standardní chybový výstup: 2.

Platí, že pokud není deskriptor uveden a prvním znakem operátoru přesměrování je <, přesmě-rování se vztahuje ke standardnímu vstupu (deskriptor 0). Je-li prvním znakem >, přesměrování se týká standardního výstupu (deskriptor 1).

Vyjasníme si to na několika praktických příkladech:

```
ls > dirlist 2>&1
```

přesměruje standardní výstup i standardní chybový výstup příkazu ls do souboru dirlist, zatímco

```
ls 2>&1 > dirlist
```

přesměruje do souboru dirlist jen standardní výstup. Tyto varianty mohou být velmi užitečné pro programátory. Oba příklady si zasluhují vyjasnění. Takže „> *někam*“ je ekvivalentní zápisu „1> *někam*“ a znamená to „přesměruj deskriptor 1 (stdout) *někam*“. Zápis „2> *někam*“ přesměruje *někam* deskriptor 2, tedy stderr. Tím máme vyjasněnu levou stranu operátoru přesměrování. Na pravé straně se objevuje „&1“, znamenající „tam kam deskriptor“. Takže přesměrování „> *někam* 2>&1“ znamená „přesměruj implicitní deskriptor 1 (stdin) do *někam* a deskriptor 2 tam, kam deskriptor 1 (tj. taky *někam*)“. Opačné pořadí zápisu (2>&1 >*někam*) způsobí, že se nejprve 2 pře-směruje do 1 (obvykle tedy /dev/console) a teprve pak se (jenom) 1 přesměruje *někam*.

Trochu se nám to komplikuje, nezaměňujte ampersand použitý v tomto kontextu s tím, jak jsme jej používali v kapitole „Interaktivní procesy“, kde sloužil ke spuštění procesu na pozadí. V tomto případě představuje pouze indikaci, že následující číslo není název souboru, ale označení deskriptoru. Všimněte si také, že mezi číslem deskriptoru a znakem „větší než“ není mezera. Pokud by tam mezera byla, bude uvedené číslo chápáno nikoliv jako deskriptor, ale opět jako název souboru. Ukazuje to následující příklad:

```
[nancy@asus /var/tmp]$ ls 2> tmp

[nancy@asus /var/tmp]$ ls -l tmp -rw-rw-r--1 nancy nancy 0 Sept 7 12:58 tmp
```

[nancy@asus /var/tmp]$ ls 2 > tmp ls: 2: No such file or directory

První zadaný příkaz je v pořádku (samotný příkaz ls ovšem nevypsal žádné chybové hlášení a soubor, do nějž byl chybový výstup přesměrován, tedy bude prázdný). Ve druhém případě je ovšem 2 chápána jako název souboru, ten neexistuje a vypíše se proto chyba.

Všechny tyto možnosti jsou podrobně popsány na informačních stránkách bashe.

Příklady

Pokud nějaký proces generuje větší množství chybových hlášení, můžete je prozkoumat takto:

příkaz 2>&1 | less

Tato možnost se často používá při překladu nových programů příkazem make, například:

andy:~/newsoft> make all 2>&1 | less ...

Oddělení standardního výstupu a standardního chybového výstupu

Následující konstrukci často používají programátoři, aby standardní výstup příkazu zobrazovali v jednom okně a jeho standardní chybový výstup v jiném okně. Nejprve příkazem tty zjistěte, který pseudoterminál používáte.

andy:~/newsoft> make all 2> /dev/pts/7

Současné zobrazení a zápis výstupu do souboru

Příkaz tee přepisuje svůj standardní vstup na standardní výstup a zároveň jej může zapisovat do jednoho či více souborů. Použijete-li volbu -a, bude k souborům připisovat. Tento příkaz je uži-tečný v případě, že chcete výstup nějakého příkazu současně sledovat a ukládat. Operátory > ani >> nedovedou provést obě tyto operace současně.

Příkaz se typicky používá prostřednictvím roury tak, jak to ukazuje následující příklad:

mireille ~/test> date | tee file1 file2 Thu Jun 10 11:10:34 CEST 2004

mireille ~/test> cat file1 Thu Jun 10 11:10:34 CEST 2004

mireille ~/test> cat file2 Thu Jun 10 11:10:34 CEST 2004

mireille ~/test> uptime | tee -a file2
  11:10:51 up 21 days, 21:21, 57 users, load average: 0.04, 0.16, 0.26
mireille ~/test> cat file2 Thu Jun 10 11:10:34 CEST 2004
  11:10:51 up 21 days, 21:21, 57 users, load average: 0.04, 0.16, 0.26

Filtry

Pokud nějaký program přijímá standardní vstup, něco s ním provádí a výsledek vypisuje na stan-dardní výstup, označuje se jako filtr. Nejčastěji se filtry používají k restrukturalizaci výstupu. Uká-žeme si několik nejběžnějších filtrů.

Více o příkazu grep

Jak už jsme viděli v kapitole „Příkaz grep“, příkaz grep prochází vstup řádek po řádku a vypisu-je ty řádky, které obsahují zadaný řetězec. Chování je možno invertovat přepínačem -v, pak se vypisují řádky, které řetězec neobsahují.

Může nás například zajímat, které soubory v aktuálním adresáři byly změněny v únoru:

jenny:~> ls -la | grep Feb

Stejně jako většina ostatních příkazů, i grep rozlišuje mezi velkými a malými písmeny. Přepína-čem -i můžete toto rozlišování potlačit. GNU verze příkazu nabízí i řadu dalších rozšíření, napří-klad volbu --color, která je užitečná ke zvýraznění hledaného řetězce v delších řádcích, nebo volbu --after-context, která nakonec vypíše celkový počet nalezených řádků. Volbou -r můžete příkaz grep spustit rekurzivně, i pro podadresáře. Jako obvykle je možné jednotlivé volby vzá-jemně kombinovat.

K přesnější specifikaci vyhledávaných textů je možné použít regulární výrazy. Nejlepší metoda, jak se s regulárními výrazy seznámit, je dokonce právě dokumentace příkazu grep. Vynikajícím způ-sobem je toto téma popsáno na informační stránce příkazu. Podrobnější vysvětlení regulárních výrazů už je mimo záběr této kapitoly, vřele vám ale doporučujeme se s nimi prostřednictvím pří-kazu grep seznámit. Výborně zpracovaný úvod do regulárních výrazů najdete například na *http://www.kai.vslib.cz/~satrapa/docs/regvyr/*.

Rozhodně se vám vyplatí si s příkazem grep nějakou chvíli pohrát, je to základní a zároveň velmi mocný filtrační příkaz. Trochu vám pomohou i cvičení v závěru této kapitoly.

Seřazení výstupu

Příkaz sort standardně seřadí řádky vstupu podle abecedy:

thomas:~> cat people-I-like | sort Auntie Emmy Boyfriend Dad Grandma Mum My boss

Tento příkaz toho ovšem dokáže mnohem více. Vezměme si například velikosti souborů. Následujícím příkazem můžete vypsat obsah adresáře a seřadit výpis od největšího k nejmenšímu souboru:

ls -la | sort -nk *5*
Volba -n zapíná numerické řazení (kde je 200>10, což u textového řazení neplatí), volba -k 5 řadí podle pátého sloupce výpisu (ve kterém příkaz ls vypisuje velikosti).

> Stará syntaxe příkazu sort Stejný výsledek můžete dostat příkazem ls -la | sort +4n, ale tato starší forma zápisu neod-povídá platným standardům.

Příkaz sort můžete použít ve spolupráci s příkazem uniq (nebo rovnou jako sort -u) a odfiltro-vat tak z výstupu duplicitní položky:

thomas:~> cat itemlist 1 4 2 5 34 567 432 567 34 555

thomas:~> sort itemlist | uniq 1 2 34 4 432 5 555 567

Shrnutí

V této kapitole jsme si ukázali, jak je možné příkazy vzájemně spojovat a jak lze výstup jednoho příkazu použít jako vstup pro další příkaz. Přesměrování vstupu a výstupu je na linuxových a unixových systémech velmi častou operací. Jde

o mocný mechanismus, který umožňuje flexibilní použití jednotlivých stavebních bloků systému. Nejčastěji používané operátory jsou > a |.

Cvičení

Následující cvičení obsahují různé příklady, jak kombinovat příkazy. Cílem je dosáhnout požadovaného výsledku s co nejmenší námahou. Všechny příkazy spouštějte jako běžný uživatel, občas tak dojde i k nějakým chybám. V takových případech nezapomeňte v manuálových stránkách zjistit, co se stalo.

- Vypište obsah nějakého dlouhého adresáře, příkazem cut z něj vyberte pouze přístupová práva souborů. Z nich prostřednictvím příkazů sort a uniq odfiltrujte duplicitní údaje. Nakonec příkazem wc vypište, kolik různých nastavení přístupových práv soubory v daném adresáři mají.

Uložte do souboru výstup příkazu date. Přidejte k němu výstup příkazu ls. Pošlete si sou-bor do své lokální e-mailové schránky (nezadávejte doménovou část adresy). Používáte-li bash, budete hned upozorněni na novou poštu.
Vypište ta zařízení v adresáři /dev, která momentálně používá vaše UID. Použijte příkaz less, abyste si výpis mohli pohodlně prohlédnout.
Jako neprivilegovaný uživatel spusťte následující příkazy. Popište, co je v jednotlivých pří-kazech standardní vstup, standardní výstup a standardní chybový výstup.
cat neexistující_soubor
file /sbin/ifconfig
grep root /etc/passwd /etc/nofiles > výsledný_soubor
/etc/init.d/sshd start > /var/tmp/output
/etc/init.d/crond start > /var/tmp/output 2>&1
Nyní si výsledky zkontrolujte tak, že příkazy spustíte znovu, přičemž standardní výstup přesměrujete do souboru /var/tmp/output a standardní chybový výstup do /var/tmp/error.
Kolik procesů v systému právě běží?
Kolik máte v domovském adresáři neviditelných souborů?
Příkazem locate najděte dokumentaci k jádru (kernel).
Zjistěte, který soubor obsahuje následující text:

> root:x:0:0:root:/root:/bin/bash

> a tento

system: root

- Co udělá tento příkaz:

  > time; date >> time; cat < time

- Jakým příkazem byste zjistili, který skript v /etc/init.d spouští určitý proces?

Textové editory

V této kapitole se budeme věnovat důležité problematice textových editorů. Zaměříme se zejmé-na na editor vim.Po prostudování kapitoly budete umět:

Otevírat a zavírat soubory v textovém režimu
Editovat soubory
Vyhledávat text
Vracet zpět provedené změny
Slučovat soubory
Obnovovat ztracené soubory
Najít program nebo balík pro kancelářské použití

Textové editory

Proč používat textový editor

Je nesmírně důležité umět používat alespoň jeden textový editor. Víte-li, jak na svém systému pou
žívat nějaký textový editor, zvyšuje se vaše nezávislost při práci s počítačem. V příští kapitole už budeme potřebovat zvládnout práci s textovým editorem, protože jím bude-me upravovat soubory, které přímo ovlivňují chování systému. Jako pokročilý uživatel budete chtít psát skripty nebo knihy, vytvářet webové stránky nebo programovat. Zvládnutím textového edi-toru vylepšíte svou produktivitu i schopnosti.

Který editor použít?

Zaměříme se na editory pracující v textovém prostředí, které totiž můžete používat i v systému bez grafického rozhraní nebo v terminálovém okně. Další výhodou takového editoru je rovněž jeho snadné použití na vzdálených počítačích. Protože přes síť nebudete přenášet celé grafické prostředí, je použití textového editoru mnohem rychlejší.

Jako obvykle existuje i v tomto případě více možností. Podívejme se na běžně dostupné editory.

Ed

Editor ed je řádkově orientovaný editor používaný k vytváření, zobrazení, úpravám a k dalším
operacím s textovými soubory, buď interaktivně nebo prostřednictvím skriptů.Editor ed je původní editor z unixových systémů, takže je k dispozici prakticky všude. Ve většiněpřípadů jej ale dnes nahrazují celoobrazovkové editory jako emacs a vi.

GNU Emacs

Emacs je rozšiřitelný, upravitelný, výborně dokumentovaný textový editor se zobrazováním v reál-ném čase, známý na většině Unixů a jiných systémů. Editovaný text je viditelný na obrazovce a automaticky se aktualizuje tak, jak uživatel zadává příkazy. Pracuje v reálném čase, protože zobrazení na displeji se aktualizuje velmi často, typicky po napsání každého znaku nebo několi-ka znaků. Tím se minimalizuje množství informací, na něž musíte při editaci myslet. Jde o pokro-čilý editor, protože nabízí celou řadu funkcí nad rámec obyčejného vkládání a mazání: Umožňu-je řízené zpracování textu, automatické odsazování kódu, současné zobrazení více souborů, edi-taci formátovaného textu a práci se znaky, slovy, řádky, větami, odstavci a stránkami, zpracovává výrazy a rozlišuje různé programovací jazyky.

V kterémkoliv okamžiku můžete stisknout Ctrl+H a dozvědět se možnosti, které v dané chvíli máte. Můžete také snadno zjistit, co který příkaz dělá nebo které příkazy se vztahují k určité ope-raci. Editor můžete upravovat a částečně tak měnit chování jeho příkazů. Pokud například použí-váte programovací jazyk, v němž komentáře začínají znaky <** a končí znaky **>, můžete Emacs naučit, aby text mezi těmito znaky chápal jako komentář. Můžete také měnit přiřazení funkčních kláves. Pokud jste například zvyklí na rozložení základních kláves pro pohyb nahoru, dolů, vlevo a vpravo do kosočtverce, můžete si klávesy tímto způsobem přemapovat.

Emacs je rozšiřitelný, protože kromě jednoduchých úprav stávajících příkazů můžete naprogra-movat vlastní příkazy. Používá se

k tomu jazyk Lisp, zpracovávaný vlastním interpreterem Emac-su. Emacs můžete rozšiřovat online – je rozdělen na celou řadu bloků, které se vzájemně volají a kterýkoliv z nich můžete v kterémkoliv okamžiku editace předefinovat. Většinu částí Emacsu je možné nahradit bez toho, abyste museli instalovat samostatnou kopii celého editoru. V Lispu je napsána většina existujících editačních příkazů, těch několik výjimek by sice rovněž mohlo být napsáno v Lispu, kvůli větší efektivitě jsou však napsány v jazyce C. Rozšíření editoru sice může vytvořit jen programátor, používat už je ale následně může kdokoliv.

Pokud Emacs spustíte v grafickém prostředí (příkazem xemacs), obsahuje nabídky příkazů a pohodlné napojení funkcí na tlačítka myši. Většinu výhod grafického prostředí však Emacs nabí-zí i v textovém terminálu. Můžete například snadno prohlížet nebo editovat několik souborů najednou, přesouvat text mezi soubory a při editaci textu spouštět příkazy shellu.

Vi(m)

Název *vim* znamená Vi IMproved. Původně to sice znamenalo Vi IMitation, ale provedených vylepšení je tolik, že je změna názvu oprávněná. Textový editor vim obsahuje prakticky všechny příkazy původního unixového editoru vi plus celou řadu dalších.

Příkazy se v editoru vi zadávají pouze prostřednictvím klávesnice, což má tu výhodu, že můžete mít ruce na klávesnici, dívat se na obrazovku a nemusíte pořád přesouvat ruku k myši. Pokud byste po tom toužili, je možno zapnout grafickou verzi s posuvníky a nabídkami.

Při editaci souborů v této knize budeme používat editor vi či vim, vy si samozřejmě můžete vybrat i jakýkoliv jiný. Rozhodně vám ale doporučujeme ovládnout alespoň základy práce s editorem vi, protože jde o standardní textový editor, který najdete prakticky na každém unixovém systému. Emacs je v řadě systémů instalován volitelně. Na různých počítačích a různých terminálech mohou být v ovládání drobné rozdíly, důležité je ale to, že pokud zvládnete práci s editorem vi, přežije-te na jakémkoliv unixovém systému.

Kromě příkazu vim je součástí balíčku také program gvim, verze editoru pro prostředí GNOME – ovšem v závislosti na distribuci to může být také balíček jiného jména, např. Vim-X11. Pro začí-nající uživatele bude tato verze jednodušší, protože pokud nevědí nebo zapomenou, jak určitou operaci provést standardními příkazy, pomůže jim systém nabídek.

Použití editoru vim

Dva režimy

Editor vi je velmi mocný nástroj s obsáhlou vestavěnou nápovědou, kterou můžete vyvolat po spuštění programu příkazem :help (v tomto případě manuálové a informační stránky selhávají, protože obsahují mnohem méně informací). Dále si popíšeme jen úplně základní operace, aby-chom vám usnadnili první kroky.

Začátečníky často mate, že editor vi pracuje ve dvou režimech – příkazovém režimu a režimu vklá-dání. Po spuštění se editor vždy nachází v příkazovém režimu. Pomocí příkazů se můžete pohy-bovat po textu, vyhledávat, nahrazovat, označovat bloky a provádět další editační úkony. Někte-rými příkazy se můžete přepnout do režimu vkládání.

Znamená to, že každá klávesa má dva různé významy – v příkazovém režimu vyvolává nějaký pří-kaz, v režimu vkládání píše odpovídající znak.

Základní příkazy Pohyb po textu

Pohyb po textu je obvykle možný klasickými kurzorovými klávesami. Pokud by nefungovaly, zkuste:

h posouvá kurzor doleva,
l posouvá kurzor doprava,
k posouvá kurzor nahoru,
j posouvá kurzor dolů.

Stiskem Shift+G se přesunete na konec dokumentu.

Základní operace

n dd smaže *n* řádků od aktuální pozice kurzoru,
n dw smaže *n* slov napravo od kurzoru,
x smaže znak, na němž se nachází kurzor,
:*n* provede přesun na *n*. řádek,
:w uloží soubor na disk,
:q ukončí editor,
:q! vynutí ukončení v případě, že jste neuložili změny,
:wq uloží soubor a ukončí editor,
:w *název* uloží soubor pod novým *názvem,*
/*řetězec* hledá v souboru *řetězec* a umístí kurzor na první výskyt za stávající pozicí kur-zoru,
/ přesune kurzor na další výskyt dříve hledaného řetězce,

1, $s/*slovo*/*jinéslovo*/g nahradí *slovo jinýmslovem*,
yy zkopíruje blok textu,
n p vloží zkopírovaný blok *n*krát,
:recover obnoví soubor po neočekávaném přerušení editoru.

Příkazy pro přepnutí do režimu vkládání

a slouží k přidání – před přepnutím do režimu vkládání posune kurzor o jeden znak vpra-vo,
i slouží ke vkládání,

- o vloží prázdný řádek pod pozici kurzoru, přesune kurzor na tento nový řádek a přepne do režimu vkládání.

Stiskem klávesy Esc se přepnete zpět do příkazového režimu. Pokud používáte hodně starou verzi vi, která v režimu vkládání nezobrazuje příznak „INSERT“, a nejste si jisti, ve kterém režimu právě jste, prostě zmáčkněte Esc a s jistotou budete v příkazovém režimu. Jestliže už v příkazovém reži-mu v té chvíli jste, po zmáčknutí Esc vás program upozorní nějakým pípnutím nebo probliknu-tím, to je normální chování.

Jednoduchý začátek

Čtení návodu je poměrně nudné, takže můžete použít příkaz vimtutor, který vás seznámí se základními příkazy vimu. Jde o asi třicetiminutový tutoriál, který popisuje většinu základních funkcí v osmi snadných lekcích. Za půl hodiny se samozřejmě nedá dozvědět o vimu všechno, měli byste se nicméně dozvědět všechno, co je k běžnému používání tohoto editoru zapotřebí.

Pokud je vim správně nainstalován, v Unixu i v MS Windows spustíte výukový text příkazem vim-tutor. Tímto příkazem se vytvoří kopie učebního textu, takže jej můžete editovat bez rizika poško-zení originálu. Výukový text existuje i v několika přeložených verzích, které volíte zadáním dvou-znakového kódu jazyka. Například francouzskou verzi spustíte (je-li nainstalována) příkazem vim-tutor *fr*. Dobře zpracovaný český návod najdete na *http://www.kai.vslib.cz/~satrapa/docs/vim/*.

Linux v kanceláři

Historie

V průběhu posledních deseti let dominuje na poli kancelářských aplikací balík MS Office. Přiznej-me si, že Microsoft Word, Excel a PowerPoint představují dnes průmyslový standard a dříve či později budete nuceni s takovými dokumenty pracovat.

Monopolní postavení Microsoftu na tomto poli představovalo velký problém při rozšiřování uživatelské základny Linuxu, proto skupina německých vývojářů zahájila projekt StarOffice, který usiloval a stále usiluje o vytvoření klonu MS Office. Koncem 90. let, těsně před vydáním verze 5.2, byla společnost StarDivision zakoupena společností Sun Microsystems. Sun sice pokračuje ve vývoji, omezil ale přístup ke zdrojovému kódu. Vývoj na původní větvi zdrojových kódů však pokračuje i v Open Source komunitě, projekt se jmenuje OpenOffice.org. Balík OpenOffice.org je k dispozici pro celou řadu platforem včetně MS Windows, Linuxu, MacOS a Solarisu. V kapitole „Deset let zkušeností k vašim službám“ se můžete podívat na obrázek této aplikace.

Současně se vyvíjí i několik dalších projektů. Další běžnou alternativou k produktům MS Office je balík KOffice, oblíbený zejména mezi uživateli distribuce SuSE. Tento klon obsahuje programy kompatibilní s MS Wordem a Excelem a řadu dalších funkcí.

Existují i menší projekty, které se zabývají jen vybranou částí balíku Office, například Abiword a MS Wordview, které usilují o kompatibilitu s dokumenty MS Word, nebo Gnumeric, kompati-bilní s excelovými tabulkami.

Balíky a programy

Moderní distribuce obvykle obsahují všechny potřebné nástroje. Součástí takových distribucí je i kvalitní dokumentace a nápověda s možností prohledávání, proto nebudeme o podobných pro-gramech hovořit podrobněji. Vše potřebné se dozvíte v dokumentaci k systému anebo na webo-vých stránkách projektů, jako jsou:

http://www.openoffice.org, http://www.openoffice.cz,
http://www.koffice.org,
Freshmeat (*http://freshmeat.net*) a Sourceforge (*http://sourceforge.org*), kde najdete řadu dalších projektů.

Poznámky Obecné používání kancelářských balíků

Snažte se omezit používání kancelářských dokumentů na prostředí, kam patří – tedy na kancelář.Příklad: Řada uživatelů Linuxu těžce nese, když jim pošlete takovýto e-mail: „Ahoj, něco pro tebemám, přečti si přílohu.“ V příloze se pak nachází wordový dokument s textem: „Tak co, jak se dařína novém místě a kdy si zajdeme spolu na oběd?“ Není také rozumné v podobných dokumentech
přikládat k e-mailu (naskenovaný) podpis. Pokud potřebujete e-mail podepsat, použijte GPG,GNU Privacy Guard, kompatibilní s

PGP.Adresát takového e-mailu není naštvaný ani tak proto, že by přiložený dokument neuměl otevřít,
ani proto, že by jej příliš trápila zbytečná velikost e-mailu. Jde o ten automatický předpoklad, ževšichni používají MS Windows, a samozřejmě také o práci navíc, spojenou se spuštěním speciál-ní aplikace.

Uživatelské a systémové konfigurační soubory

V další kapitole začneme měnit nastavení operačního systému, což bude zahrnovat editaci různých souborů, které definují chování různých programů.

Nikdy tyto soubory needitujte pomocí kancelářského balíku!

Kancelářské aplikace přidávají do editovaných textů vlastní informace o formátování a dalších vlastnostech textu. Program, který se má daným konfiguračním souborem řídit, ale nebude těmto údajům navíc rozumět a výsledkem bude, že program bude fungovat špatně, nebo dokonce vůbec. Můžete samozřejmě soubor explicitně ukládat jako neformátovaný text, je to ale zbytečná práce navíc.

Já ale chci grafický editor!

Pokud na tom doopravdy trváte, vyzkoušejte gedit, kedit, kwrite nebo xedit. Tyto programy pra-cují s čistými textovými soubory, což je to, co potřebujeme. Ve Windows jim přibližně odpovídá „Poznámkový blok“ – uvedené programy jsou nicméně chytřejší přinejmenším o takové věci, jako je zvýraznění syntaxe. Pokud byste chtěli dělat něco seriozního, pak použijte pořádný textový edi-tor, například vim nebo emacs.

Shrnutí

V této kapitole jsme se naučili používat textový editor. Volba konkrétního editoru je samozřejměvěcí osobní volby, alespoň jeden je však nutné znát.Na všech unixových systémech je k dispozici editor vi.
Většina linuxových distribucí obsahuje balík kancelářských aplikací a textový editor s grafickým rozhraním.

Cvičení

V této kapitole je pouze jediné cvičení: Příkazem vimtutor otevřete výukový text vimu a začněte.Druhou možností je spustit emacs a následně stiskem Ctrl+H a T jeho výukový text.Jediný možný způsob je praxe!

Home sweet /home

V této kapitole budeme hovořit o konfiguraci uživatelského prostředí. Nyní už víme, jak zacházet s editorem, takže můžeme měnit různé soubory tak, abychom se doma cítili co nejlépe. Po pro-studování této kapitoly budete vědět více o následujících tématech:

Organizace prostředí.
Základní konfigurační soubory shellu
Nastavení shellu
Nastavení výzvy
Nastavení grafického prostředí
Zvukové aplikace a video aplikace
Správce displeje a oken
Jak pracuje klient-server systém X Window
Nastavení jazyků a fontů
Instalace nových programů
Aktualizace nainstalovaných balíčků

Obecné zásady organizace pracovního prostředí

Úvod

Jak už jsme říkali, je poměrně jednoduché vyrobit v systému zmatek. Proto zdůrazňujeme, že je velmi důležité udržovat si v systému pořádek. Pokud budete mít toto pravidlo na paměti hned od počátku, zvyknete si na ně a ušetříte si spoustu času, až budete v linuxovém nebo unixovém systému programovat nebo až budete takový systém spravovat. Uveďme si několik základních pravidel, která vám usnadní život:

Skripty a programy ukládejte v samostatném adresáři bin.

Nespustitelné soubory ukládejte ve vhodných adresářích, vytvořte si tolik adresářů, kolik uznáte za vhodné. Používejte například samostatné adresáře pro obrázky, dokumenty, pro-jekty, stažené soubory, tabulky, osobní soubory a podobně.
Příkazem chmod *700* adresář skryjete adresář před ostatními uživateli.
Názvy souborů volte tak, aby byly přiměřeně informativní, například Stiznost na urad vlady 20060415 místo dopis1. Obecně je také vhodné (z důvodů přenositelnosti mezi různými platformami) nepoužívat v názvech českou diakritiku.

Udělejte si místo

Na některých systémech vás nastavené diskové kvóty donutí čas od času uklízet, případně může-te narazit na limity dané fyzickou kapacitou disku. Budeme hovořit o různých způsobech, jak na disku získat volné místo jinak než příkazem rm.

Příkazem quota -v zjistíte, kolik místa ještě můžete využít. Tento příkaz použijte, jen pokud jste omezeni diskovými kvótami. Pokud vás omezuje velikost disku, použijte příkaz df -h ..

Vyprázdnění souboru

V některých případech vás nemusí zajímat samotný obsah souboru, postačí vám jen jeho název jako značka (stačí vám například čas vzniku souboru jako připomenutí, že tento soubor v určité době existoval, nebo chcete jen naznačit, že takový soubor budete muset v budoucnu vytvořit). Prázdný soubor vytvoříte (a existující vyprázdníte) přesměrováním prázdného příkazu. V Bourne a Bash shellu to vypadá takto:

andy:~> cat wishlist > placeholder andy:~> ls -la placeholder -rw-rw-r--1 andy andy 200 Jun 12 13:34 placeholder

andy:~> > placeholder andy:~> ls -la placeholder -rw-rw-r--1 andy andy 0 Jun 12 13:35 placeholder

Operace, při níž existující soubor zkrátíme na nulovou délku, se označuje jako „truncating“.Pokud chcete vytvořit nový prázdný soubor, dosáhnete stejného efektu příkazem touch. Pokud
stejnojmenný soubor existuje, příkaz touch pouze „sáhne“ na čas jeho poslední aktualizace.Podrobnější údaje naleznete na informační stránce příkazu touch.Chcete-li soubor „skoro vyprázdnit“, použijte příkaz tail. Řekněme, že *andy* už má docela dlouhý
seznam přání, protože v souboru stále připisuje a připisuje a nikdy z něj nic nemaže. Teď bude
chtít ponechat jenom posledních pět řádků:

andy:~> tail -5 wishlist > newlist andy:~> cat newlist > wishlist andy:~> rm
newlist

Více o logovacích souborech

Některé linuxové programy zapisují do logovacích souborů všechny možné informace. Většinou existuje způsob, jak nastavit, aby program zaznamenával jenom chyby či obecně co nejméně infor-mací – mění se například míra podrobnosti vypisovaných údajů. I tak ale logovací soubory stále narůstají a je nutné se jim věnovat. Nabízíme několik způsobů, jak se logovacích souborů úplně zbavit, nebo alespoň rozumně omezit jejich velikost:

Pokud program neběží a jste si jisti, že jeho záznamy nebudete potřebovat, logovací sou-bor smažte. Některé programy při dalším startu zjistí, že logovací soubor neexistuje, a už jej nevytvoří.
Pokud logovací soubor smažete a program jej znovu vytvoří, přečtěte si dokumentaci k programu a zkuste zjistit, jak logování vypnout.
Omezte velikost logovacích souborů tím, že budete zaznamenávat jenom údaje, které vás zajímají nebo které jsou důležité.
Zkuste nahradit logovací soubor symbolickým odkazem na /dev/null; při troše štěstí to programu nebude vadit. Nedělejte to u programů, které se spouštějí při bootování systému nebo automaticky cronem (viz kapitolu „Procesy“). Tyto programy mohou symbolický odkaz přepsat skutečným souborem, který začne znovu narůstat.

Pošta

Pravidelně promazávejte poštovní schránku, vytvořte si v ní složky a nastavte automatické třídění pošty programem procmail (viz informační stránky) nebo přímo v používaném poštovním klien-tovi. Pokud poštovní klient vytvoří „odpadkový koš“, pravidelně jej promazávejte.

Přesměrování pošty můžete v domovském adresáři nastavit souborem .forward. Linuxové poštov-ní služby při lokálním doručování pošty vždy kontrolují obsah tohoto souboru. V něm můžete nastavit, co se má s poštou dělat. Může obsahovat jediný řádek, na kterém bude zapsána úplná e-mailová adresa. V takovém případě systém veškerou poštu přepošle na tuto adresu. Pokud máte například pronajatý prostor na webovém serveru, můžete takto přesměrovat veškerou poštu pro účet *webmaster* na svůj vlastní účet. Soubor .forward může vypadat například takto:

webmaster@www ~/> cat .forward mike@pandora.be

Vhodně nastavené přesměrování pošty vám také ušetří práci s kontrolou více poštovních schrá
nek. Veškeré účty si můžete přesměrovat na jeden centrální, snadno přístupný účet. Můžete také požádat správce systému, aby vám přesměrování pošty nastavil v systémovém sou-boru aliasů – to se používá například v situacích, kdy se ruší uživatelský účet, potřebujete ještě ale nějakou dobu zajistit doručování pošty.

Šetření místem pomocí odkazů

Pokud s jedním programem nebo souborem potřebuje pracovat více uživatelů nebo pokud je název souboru příliš dlouhý či příliš těžko zapamatovatelný, nekopírujte soubory fyzicky a vytvořte místo toho symbolické odkazy.

Více symbolických odkazů na stejný soubor může mít různé názvy, takže v adresáři jednoho uži-vatele se odkaz může jmenovat monfichier, v adresáři jiného uživatele se může jmenovat mylink. Můžete také vytvořit více symbolických odkazů s různými názvy v jednom adresáři. Velmi časté je to v adresáři /lib. Zkuste zadat příkaz:

```
ls -l /lib
```

Uvidíte, že adresář obsahuje řadu odkazů na jeden a ten samý soubor. Dělá se to proto, aby se vyhovělo všem programům, které určitou knihovnu hledají pod různými názvy. Tímto způsobem vám stačí jedna instance knihovny pro všechny.

Omezení velikosti souborů

Shell obsahuje vestavěný příkaz pro omezení velikosti souborů, ulimit, můžete jej také použít k výpisu údajů o nastavených omezeních systémových prostředků:

```
cindy:~> ulimit -a

core file size (blocks)         0
data seg size (kbytes)          unlimited
file size (blocks)              unlimited
max locked memory (kbytes)      unlimited
max memory size (kbytes)        unlimited
open files                      1024

pipe size (512 bytes)           8
stack size (kbytes)             8192
cpu time (seconds)              unlimited
max user processes              512
virtual memory (kbytes)         unlimited
```

Cindy není programátor, a proto ji nezajímají „core dumpy“, což jsou speciální soubory, které obsahují údaje pro potřeby ladění havarovaných programů. Pokud tyto soubory potřebujete, nastavte jejich velikost příkazem ulimit. Podrobnosti naleznete na informačních stránkách k bashi.

Komprimované soubory

Komprimované soubory jsou velmi užitečné, protože na disku zabírají méně místa. Rychleji se také přenášejí po síti. Celá řada souborů je v systému uložena v komprimované podobě – například manuálové stránky. Bylo by ovšem docela zdlouhavé pokaždé, když potřebujeme zjistit nějakou informaci, soubor rozbalovat a následně znovu zabalovat. Určitě nemáte chuť rozbalit manuálo-vou stránku, přečíst si dvě věty o nějakém parametru a následně stránku znovu zabalit. Většina lidí by navíc v takové situaci určitě zapomněla na opětovné zabalení.

Existují proto nástroje, které pracují přímo s komprimovanými soubory a rozbalují je pouze v paměti. Na disku zůstává soubor ve své komprimované podobě. Na většině systémů najdete pří-kazy jako zgrep, zcat, bzless a podobné, při jejichž použití se vyhnete zbytečné dekomprimaci a opakované komprimaci souborů. Podívejte se do adresáře binárních souborů a na příslušné informační stránky.

O samotné komprimaci souborů a příkladech vytváření archivů budeme hovořit v kapitole „Základní techniky zálohování“.

Textové prostředí

Proměnné prostředí Obecné informace

Už dříve jsme se zmiňovali o některých proměnných prostředí, například o proměnných PATH a HOME. Doposud jsme si jen ukazovali příklady, jak tyto proměnné použít k určitému účelu. Existuje ovšem celá řada programů a nástrojů, které ke své činnosti potřebují mnoho různých informací o vás a vašem prostředí.

Co dalšího kromě vyhledávacích cest a umístění domovského adresáře mohou ještě programy

potřebovat? Řada programů chce vědět, jaký terminál používáte – tyto informace jsou uloženy v proměnné TERM. V textovém režimu se používá emulátor terminálu *linux*, v grafickém režimu s největší pravděpodobností používáte *xterm*. Mnoho programů také zajímá, jaký je váš oblíbený textový editor pro případ, že budou potřebovat spustit editor jako svůj podřízený proces. Název používa-ného shellu je uložen v proměnné SHELL, typ operačního systému v proměnné OS a tak dále. Celý seznam všech proměnných prostředí můžete získat příkazem printenv.

Proměnné prostředí spravuje shell. Na rozdíl od normálních proměnných shellu dědí proměnné prostředí každý spouštěný program včetně dalších instancí shellu. Nové procesy získají vlastní kopii těchto proměnných, kterou mohou číst, modifikovat a dále předávat svým synovským procesům.

Pro názvy proměnných prostředí neplatí žádná speciální pravidla, existuje pouze zavedená konvence, že se zapisují velkými písmeny. Můžete si zavést jakékoliv vlastní názvy, některé standardní proměnné, jako například PATH a HOME, jsou ovšem natolik důležité, že se na všech systémech jmenují stejně.

Export proměnných

Obsah konkrétní proměnné typicky zjistíte příkazem echo, jak ukazují následující příklady:

```
debby:~> echo $PATH /usr/bin:/usr/sbin:/bin:/sbin:/usr/X11R6/bin:/usr/local/bin

debby:~> echo $MANPATH /usr/man:/usr/share/man/:/usr/local/man:/usr/X11R6/man
```

Pokud budete chtít obsah proměnných změnit tak, aby bylo nové nastavení přístupné ostatním programům, musíte provést export nové hodnoty ze svého prostředí do prostředí, z nějž se ostat-ní programy spouštějí. Typickým příkladem je export proměnné PATH. Budete-li například chtít spouštět letecký simulátor z adresáře /opt/FlightGear/bin, můžete novou hodnotu nastavit takto:

```
debby:~> PATH=$PATH:/opt/FlightGear/bin
```

Tím shellu říkáte, aby programy hledal nejen v aktuálně nastavené cestě, $PATH, ale navíci v adresáři /opt/FlightGear/bin.Dokud ovšem prostředí nebude znát novou hodnotu proměnné PATH, nebude to fungovat:

```
debby:~> runfgfs bash: runfgfs: command not found
```

Export nové hodnoty proměnné provedete vestavěným příkazem export:

```
debby:~> export PATH

debby:~> runfgfs --spouští se letecký simulátor--
```

V bashi to typicky provedeme jedním elegantním krokem:

export PROMĚNNÁ=*hodnota*

Stejný způsob se používá u proměnné MANPATH, která příkazu man říká, kde má hledat kom-primované manuálové stránky. Pokud nainstalujete nový program do nového nebo netypického adresáře, dokumentace se nejspíš rovněž nainstaluje do nějakého neobvyklého adresáře. Budete -li si chtít přečíst manuálové stránky nového programu, musíte vyexportovat novou proměnnou MANPATH:

```
debby:~> export MANPATH=$MANPATH:/opt/FlightGear/man

debby:~> echo $MANPATH /usr/man:/usr/share/man:/usr/local/man:/usr/X11R6/man:/opt/FlightGear/man
```

Opakovanému nastavování nové hodnoty ve všech otevřených oknech se můžete vyhnout tím, že novou hodnotu nastavíte v některém z konfiguračních souborů shellu, viz kapitolu „Export proměnných“.

Rezervované proměnné

Následující tabulka obsahuje přehled nejčastěji používaných proměnných: Řada proměnných je nejen předdefinována, ale prostřednictvím konfiguračních souborů má i něja-kou přednastavenou hodnotu. O tom budeme hovořit v následující části.

Název proměnné	Uložená hodnota
DISPLAY	Používá se systémem X Window k identifikaci displeje.
DOMAIN	Název domény.
EDITOR	Výchozí textový editor.
HISTSIZE	Počet řádků uložených v historii příkazů shellu.
HOME	Cesta k domovskému adresáři.
HOSTNAME	Název lokálního systému.

INPUTRC	Umístění definičních souborů vstupních zařízení, obsahuje například mapování klávesnice ve specifických situacích.
LANG	Preferovaný jazyk.
LD_LIBRARY_PATH	Vyhledávací cesta pro knihovny.
LOGNAME	Přihlašovací jméno.
MAIL	Umístění složky příchozí pošty.
MANPATH	Vyhledávací cesta pro manuálové stránky.
OS	Řetězec popisující operační systém.
OSTYPE	Další podrobnosti o verzi OS a podobně.
PAGER	Používá se v programech jako v situaci, kdy je výstup delší než jedna stránka.
PATH	Vyhledávací cesta pro příkazy.
PS1	Primární prompt.
PS2	Sekundární prompt.
PWD	Aktuální pracovní adresář.
SHELL	Aktuální shell.
TERM	Typ terminálu.
UID	Identifikátor uživatele.
USER(NAME)	Jméno uživatele.
VISUAL	Přednastavený celoobrazovkový editor.
XENVIRONMENT	Umístění osobních nastavení systému X Window.
XFILESEARCHPATH	Vyhledávací cesta pro grafické knihovny.

Běžné proměnné prostředí

Konfigurační soubory shellu

Spustíte-li příkaz ls -al, vypíší se všechny soubory v aktuálním adresáři včetně souborů, jejichž název začíná tečkou. V domovském adresáři si můžete všimnout několika souborů, jejichž název začíná tečkou a končí znaky *rc*. Bash například používá soubor .bashrc. Jde o uživatelský protěj-šek systémového (globálního) konfiguračního souboru /etc/bashrc.

Když se interaktivně přihlašujete k systému, provede program login vaši autentizaci, nastaví pro-středí a spustí shell. V případě shellu bash se při jeho spouštění nejprve načítá obecný konfigu-rační soubor profile z adresáře /etc, samozřejmě pokud existuje. Následně bash hledá soubory
~/.bash_profile, ~/.bash_login a ~/.profile v tomto pořadí. První nalezený (a čitelný) přečte a pro-vede v něm uvedené příkazy.
Pokud nenalezne žádný z nich, provede příkazy ze souboru/etc/bashrc.
Když se odhlašujete, bash přečte a provede příkazy ze souboru ~/.bash_logout, samozřejměpokud soubor existuje.Celý proces je podrobně popsán na manuálových stránkách login a bash.

Typická sestava konfiguračních souborů

Zkusme se na některé z těchto souborů podívat. Prvním načítaným je soubor /etc/profile, ve kte-rém se nastavují důležité proměnné jako PATH, USER a HOSTNAME:

```
debby:~> cat /etc/profile # /etc/profile

# System wide environment and startup programs, for login setup # Functions and aliases go in /etc/bashrc


  # Path manipulation if [ `id -u` = 0 ] && ! echo $PATH | /bin/grep -q
  "/sbin" ; then PATH=/sbin:$PATH fi


   if [ `id -u` = 0 ] && ! echo $PATH | /bin/grep -q "/usr/sbin" ; then PATH=/usr/sbin:$PATH fi

if [ `id -u` = 0 ] && ! echo $PATH | /bin/grep -q "/usr/local/sbin" then PATH=/usr/local/sbin:$PATH
fi
```

```
if ! echo $PATH | /bin/grep -q "/usr/X11R6/bin" ; then
        PATH="$PATH:/usr/X11R6/bin" fi
```

Tato část skriptu kontroluje nastavení vyhledávací cesty: Pokud shell otevřel uživatel *root* (jeho UID je 0), kontroluje se, zda jeho cesta obsahuje adresáře /sbin, /usr/sbin a /usr/local/sbin. Pokud ne, příslušný adresář se přidá. U všech uživatelů se kontroluje, zda mají v cestě adresář /usr/X11R6/bin.

```
# No core files by default ulimit -S -c 0 > /dev/null 2>&1
```

Zakazují se coredumpy a výstup příslušného příkazu se přesměruje do /dev/null.

```
USER=`id -un`
LOGNAME=$USER
MAIL="/var/spool/mail/$USER"


HOSTNAME=`/bin/hostname`
HISTSIZE=1000
```

Zde se nastavují hodnoty všech obecných proměnných.

```
if [ -z "$INPUTRC" -a ! -f "$HOME/.inputrc" ]; then
        INPUTRC=/etc/inputrc fi
```

Pokud není nastavena hodnota proměnné INPUTRC a v domovském adresáři uživatele neexistu-je soubor .inputrc, nastaví se proměnná z celosystémového konfiguračního souboru vstupních zařízení.

```
export PATH USER LOGNAME MAIL HOSTNAME HISTSIZE INPUTRC
```

Všechny proměnné se exportují, aby je mohly používat ostatní programy.

```
            for i in /etc/profile.d/*.sh ; do
            if [ -r $i ]; then
            . $i
        fi done unset i
```

Přečtou se a spustí všechny čitelné soubory z adresáře /etc/profile.d. Tímto způsobem se napří-klad zapíná *color-ls*, nastaví se alias vi na vim, nastaví se hodnoty národního prostředí a podob-ně. Pomocná proměnná i se nakonec zruší, aby neovlivňovala další chování shellu.

Poté bash zpracovává soubor .bash_profile v domovském adresáři uživatele:

```
debby:~> cat .bash_profile ######################################################################
#                                                                              #
#    .bash_profile file                                                        #
#                                                                              #
#    Executed from the bash shell when you log in.                             #
#                                                                              #

##############################################################################
```

source ~/.bashrc source ~/.bash_login

Tento velmi jednoduchý soubor říká, že se má nejprve zpracovat soubor ~/.bashrc a následně sou-bor ~/.bash_login. S vestavěným příkazem source se při práci v shellu potkáte velmi často. Slou-ží k uplatnění změn konfigurace v aktuálním prostředí. Přesněji řečeno, příkaz source provede zadaný skript v rámci aktuálního shellu a nespouští jej v novém subshellu.

V souboru ~/.bash_login se příkazem umask definuje výchozí nastavení souborových práv, viz kapitolu „Souborová maska“. V souboru ~/.bashrc se nastavují různé uživatelské aliasy a funkce a osobní proměnné prostředí. Nejprve se načte soubor /etc/bashrc, v němž se definuje výchozí prompt (PS1) a výchozí hodnota umask. Pak můžete dodat vlastní nastavení. Pokud soubor ~/.bashrc neexistuje, automaticky se načte /etc/bashrc:

```
debby:~> cat /etc/bashrc# /etc/bashrc

# System wide functions and aliases
# Environment stuff goes in /etc/profile


# by default, we want this to get set.
# Even for non-interactive, non-login shells.
if [ `id -gn` = `id -un` -a `id -u` -gt 99 ]; then

                umask 002 else
                umask 022 fi

        # are we an interactive shell? if [ "$PS1" ]; then if [ -x /usr/bin/tput ];
        then if [ "x`tput kbs`" != "x" ]; then # We can't do this with "dumb"
        terminal stty erase `tput kbs` elif [ -x /usr/bin/wc ]; then if [ "`tput kbs|
        wc -c `" -gt 0 ]; then # We can't do this with "dumb" terminal stty erase
        `tput kbs` fi
         fi
    fi
    case $TERM in
            xterm*) if [ -e /etc/sysconfig/bash-prompt-xterm ]; then
            PROMPT_COMMAND=/etc/sysconfig/bash-prompt-xterm else
            PROMPT_COMMAND='echo -ne "\033]0;${USER}@$
            {HOSTNAME%%.*}:\ ${PWD/$HOME/~}\007"' fi ;; *) [
            -e /etc/sysconfig/bash-prompt-default ] &&
            PROMPT_COMMAND=\ /etc/sysconfig/bash-prompt-default
                    ;;
        esac
        [ "$PS1" = "\\s-\\v\\\$ " ] && PS1="[\u@\h \W]\\$ "

                if [ "x$SHLVL" != "x1" ]; then # We're not a login
                        shell for i in /etc/profile.d/*.sh; do if [ -x $i ];
                        then . $i fi done fi fi
```

Při odhlášení se spouštějí příkazy ze souboru ~/.bash_logout, kde se může například vymazat obrazovka terminálu:

```
debby:~> cat .bash_logout # ~/.bash_logout

clear
```

V další části si podrobněji popíšeme, jak tyto skripty pracují. Mějte po ruce příkaz info *bash*.

Prompt bashe Úvod

Prompt bashe toho dokáže mnohem víc, než jen zobrazit informace o uživatelském jméně, názvu počítače a názvu aktuálního adresáře. Můžeme do něj přidat i další údaje, například datum a čas, počet právě přihlášených uživatelů a podobně.

Než ale začneme, raději si aktuální nastavení promptu uložíme do jiné proměnné:

```
[jerry@nowhere jerry]$ MYPROMPT=$PS1

[jerry@nowhere jerry]$ echo $MYPROMPT [\u@\h \W]\$

[jerry@nowhere jerry]$
```

Když nyní změníme prompt, například příkazem PS1="->", vždy se budeme moci vrátit k původ-nímu nastavení příkazem PS1= $MYPROMPT. Samozřejmě dokud budeme prompt měnit jen na příkazovém řádku a nebudeme zasahovat do konfiguračních souborů, obnoví se původní nasta-vení i po novém přihlášení.

Několik příkladů

Vysvětlení jednotlivých nastavení a používaných speciálních sekvencí naleznete v manuálových nebo informačních stránkách bashe.

- export PS1="*[\t \j] *"

  Zobrazí čas a počet spuštěných úloh.

- export PS1="*[\d][\u@\h \w] : *"

Zobrazí datum, jméno uživatele, název počítače a aktuální pracovní adresář. Sekvence \W by zobrazila jen poslední část názvu pracovního adresáře.

- export PS1=*"{\!} "*

  Zobrazí číslo záznamu v souboru historie.

- export PS1=*"\[\033[1;35m\]\u@\h\[\033[0m\] "*

  Zobrazí text *uživatel@počítač* růžovou barvou.

- export PS1=*"\[\033[1;35m\]\u\[\033[0m\] \[\033[1;34m\]\w\[\033[0m\] "*

  Zobrazí jméno uživatele růžově a aktuální adresář modře.

- export PS1=*"\[\033[1;44m\]$USER is in \w\[\033[0m\] "*

  Nastavení „vhodné“ pro uživatele, kteří nedokážou rozlišit mezi promptem a tím, co píšou.

- export PS1=*"\[\033[4;34m\]\u@\h \w \[\033[0m\]"*

  Podtržený prompt.

- export PS1=*"\[\033[7;34m\]\u@\h \w \[\033[0m\] "*

  Bílé znaky na modrém pozadí.

- export PS1=*"\[\033[3;35m\]\u@\h \w \[\033[0m\]\a"*

  Světle růžový prompt.

- export PS1=...

Proměnnou exportujeme proto, aby dále spouštěné příkazy věděly o změně prostředí. Preferova
né nastavení promptu si můžete uložit nejlépe do souboru ~/.bashrc.Pokud chcete, může prompt dokonce spouštět shellové skripty a za různých okolností se chovatrůzně. Po každém zadání příkazu může prompt zahrát i nějakou tu melodii, i když to vás prav-děpodobně velmi rychle přejde. Další informace naleznete v dokumentu Bash-Prompt HOWTO, *http://www.tldp.org/HOWTO/Bash-Prompt-HOWTO/*.

Shellové skripty Co jsou to skripty?

Jak už jsme viděli v příkladech konfiguračních souborů, shellový skript je textový soubor, který obsahuje příkazy shellu. Pokud takovýto soubor předáte jako parametr při spouštění bashe a záro-veň nepoužijete volby -c ani -s, bash přečte a provede příkazy zapsané v souboru a skončí. Tomu-to režimu práce se říká neinteraktivní shell. Když bash zpracovává skript, nastavuje speciální para-metr 0, který obsahuje název souboru (a nikoliv název shellu, jak je tomu jindy), a následně další poziční parametry odpovídající dalším parametrům zadaným za názvem skriptu. Pokud takové parametry nejsou, příslušné proměnné se nenastaví.

Pokud příkazem chmod nastavíte příznak spustitelnosti, změní se shellový skript ve spustitelný sou-bor. Pokud takový soubor uložíte do některého z adresářů v prohledávací cestě a zadáte jeho název, Bash jej provede prostřednictvím subshellu, který pro tento účel spustí. Jinak řečeno, příkaz:

soubor *PARAMETRY*

je ekvivalentní příkazu:

bash *soubor PARAMETRY*

samozřejmě za předpokladu, že „soubor“ je spustitelný shellový skript. Subshell provádí svou rei-nicializaci, takže efekt je stejný, jako by byl skript interpretován úplně novým shellem s tou výjim-kou, že subshell od rodičovského shellu zdědí zapamatovaná umístění příkazů (viz informační stránky příkazu hash).

Většina verzí Unixu používá následující mechanismus spouštění skriptů. Pokud první řádek skrip-tu začíná dvojicí znaků „#!“, pak se zbytek řádku chápe jako název interpretru daného skriptu. Můžete tedy jako interpretr nadefinovat bash, awk, perl nebo jakýkoliv jiný program a zbytek skriptu napsat v odpovídajícím jazyce.

Interpretru budou předány následující parametry: Nejprve nepovinný parametr uvedený za názvem interpretru na prvním řádku skriptu, dále název skriptu a zbytek parametrů, s nimiž byl skript spuštěn. Pokud tuto akci nezajistí přímo operační systém, provede to za něj bash.

Skripty psané pro bash obvykle začínají řádek „#!/bin/bash“ (samozřejmě za předpokladu, že je bash nainstalován v adresáři /bin), čímž se zajistí, že skript bude interpretován bashem i v přípa-dě, že jej uživatel spouští z jiného shellu.

Několik jednoduchých příkladů

Velmi jednoduchý skript, který uživatele pouze pozdraví:

[jerry@nowhere ~] cat hello.sh #!/bin/bash echo "Hello $USER"

Tento skript obsahuje pouze jediný příkaz, echo, který uživatele pozdraví jeho jménem, což je *hodnota* ($) uložená v proměnné USER.
Následující skript je už o něco delší, používám jej pro vytvoření záložních kopií všech souborů v adresáři. Skript nejprve do proměnné LIST uloží seznam všech souborů v adresáři. Následně pro všechny soubory v tomto seznamu vytvoří název záložního souboru, zkopíruje soubor a vypíše název zkopírovaného souboru:

```
tille:~> cat bin/makebackupfiles.sh #!/bin/bash # make copies of all files in a directory LIST=`ls` for i in $LIST; do
        ORIG=$i
        DEST=$i.old
        cp $ORIG $DEST
        echo "copied $i"

done
```

Prostý příkaz typu cp * **.old* nebude fungovat, což můžete zjistit na vhodné skupině testovacích souborů. Příkaz echo je do skriptu přidán proto, aby bylo vidět, že probíhá nějaká aktivita. Obec-ně je příkaz echo velmi užitečný při ladění nefungujících skriptů: Přidejte jej za každý podezíra-ný krok a velmi rychle zjistíte, kde je chyba.

Řadu příkladů různých skriptů naleznete v adresáři /etc/rc.d/init.d. Podívejme se na následující skript, který řídí fiktivní server služby ICanSeeYou:

```
#!/bin/sh # description: ICanSeeYou allows you to see networked people

# process name: ICanSeeYou # pidfile: /var/run/ICanSeeYou/ICanSeeYou.pid # config: /etc/ICanSeeYou.cfg

# Source function library. . /etc/rc.d/init.d/functions

# See how (with which arguments) we were called. case "$1" in
        start)
                echo -n "Starting ICanSeeYou: "
                daemon ICanSeeYou
                echo
                touch /var/lock/subsys/ICanSeeYou
                ;;

        stop)
                echo -n "Shutting down ICanSeeYou: "
                killproc ICanSeeYou
                echo
                rm -f /var/lock/subsys/ICanSeeYou
                rm -f /var/run/ICanSeeYou/ICanSeeYou.pid
                ;;

        status)
                status ICanSeeYou
                ;;

        restart)
                $0 stop

                $0 start
                ;;
        *)
                echo "Usage: $0 {start|stop|restart|status}"
                exit 1 esac

exit 0
```

Nejprve se příkazem . (tečka) nahraje sada funkcí, které používají prakticky všechny skripty v adresáři /etc/rc.d/init.d. Pak následuje příkaz case, který definuje čtyři různé způsoby, jak může být skript spuštěn. Jedna možnost je například ICanSeeYou *start*. K rozhodnutí o tom, jakým způsobem byl skript volán, použijete první parametr skriptu, jehož hodnotu přečteme výrazem *$1*.

Pokud není zadán žádný ze známých způsobů spuštění, provede příkaz case implicitní akci, defi-novanou hvězdičkou, která vypíše chybové hlášení. Seznam variant příkazu case končí příkazem esac. V případě spouštění služby bude příslušný program

spuštěn jako démon, uloží se PID pro-cesu a vytvoří se zámek. V případě ukončování služby dojde k zastavení procesu a smazání ulo-ženého PID a zámku. Používané volby jako daemon a funkce jako killproc jsou definovány právě v souboru /etc/rc.d/init.d/functions. Uspořádání skriptu je specifické pro každou distribuci. Inici-alizační skripty vaší distribuce mohou používat jinou sadu funkcí definovanou v jiném souboru, případně nemusí používat žádné funkce.
V případě úspěchu vrátí skript svému rodiči návratový kód nula.Uvedený skript je pěkným příkladem použití funkcí, které zvyšují čitelnost skriptů a usnadňujíjejich tvorbu. Všimněte si také, že používají shell sh místo bash, aby je bylo možné použít co nej-univerzálněji. Na linuxovém systému spustí příkazy sh i bash shell v režimu kompatibilním s nor-mou POSIX.Manuálové stránky bashe obsahují další informace o kombinacích příkazů, smyčkách for a while,
regulárních výrazech a samozřejmě na nich najdete i příklady. Příručku o bashi vhodnou pro správce systému a pokročilé uživatele, doplněnou také cvičeními, od stejného autora jako tato pří-ručka, najdete na adrese *http://tille.xalasys.com/training/bash/*. Podrobný popis všech funkcí bashe a jejich použití pak najdete v referenční příručce Advanced Bash Scripting, *http://tldp.org/LDP/abs/html/index.html*. Velmi dobré informace obsahuje i třetí část této knihy – „Bash pro začátečníky“.

Grafické prostředí

Úvod

Průměrný uživatel se možná nebude příliš zajímat o svá přihlašovací nastavení, ovšem Linux nabí-zí i řadu úžasných správců oken a správců pracovní plochy, které se používají v grafickém pro-středí X Window. Použití i nastavení správce oken a plochy je snadné, a lze dokonce nastavit vzhled připomínající standardní prostředí MS Windows, Macintosh nebo UNIX CDE, ovšem větši-na uživatelů Linuxu dává přednost propracovaněji nastaveným pracovním plochám. Způsobům nastavení plochy se zde nebudeme věnovat. Prostě si to vyzkoušejte a přečtěte si dokumentaci ve vestavěné nápovědě, kterou tyto systémy nabízejí.

Podrobněji se ovšem podíváme na vnitřní fungování grafického subsystému.

Systém X Window

Systém X Window, často též označovaný jen jako „X“ nebo „X11“, je síťově transparentní okenní systém, který může běžet na celé řadě platforem. Server systému X distribuuje klientským progra-mům uživatelský vstup a přijímá od nich výstupní požadavky, přičemž k tomu používá různé mechanismy komunikace mezi procesy. Nejběžnější situace je, že klienti i server běží na stejném stroji, nicméně klienti mohou naprosto transparentně běžet i na jiných počítačích s jinou archi-tekturou a operačním systémem. Jak to provést, to se dozvíme v kapitole věnované sítím a vzdá-leným aplikacím.

X Window podporuje překrývající se hierarchická okna a operace s textem a grafikou, na mono-chromatickém i barevném displeji. Klientských programů, které používají služby X serveru, exis-tuje celá řada. Přímo v distribuci X Window najdete mimo jiné tyto:

xterm – emulátor terminálu,
twm – minimalistický správce oken,
xdm – správce displeje,
xconsole – program pro přesměrování konzoly,
bitmap – editor bitmap,
xauth, xhost a iceauth – programy pro řízení přístupu,
xset, xmodmap a řada dalších – programy pro změnu uživatelských nastavení,
xclock – hodiny,
xlsfonts a další – zobrazovač fontů, nástroje pro zjištění informací o fontech,
xfs – fontserver
a další...

Podrobnější informace k jednotlivým programům naleznete samozřejmě na příslušných manuálo-vých stránkách. Podrobnější popis dostupných funkcí se nachází v dokumentu *Xlib – C language X Interface*, který je součástí distribuce X, ve specifikaci *X Window System Protocol* a v manuálech a dokumentacích různých nástrojů. Odkazy na tyto a celou řadu dalších dokumentů naleznete v adresáři /usr/share/doc.

Jako uživatelské příspěvky distribuce X Consortium a na celé řadě anonymních FTP serverů může-te najít mnoho dalších nástrojů, správců oken, her a doplňků. Dobrým místem pro zahájení hle-dání jsou adresy *http://www.x.org* a *http://www.xfree.org,* případně softwarové repozitáře vaší linu-xové distribuce.

Klientem X serveru jsou samozřejmě také všechny ostatní grafické aplikace, například webový prohlížeč, poštovní program, prohlížeč obrázků, přehrávač videa a podobně. Při normálním způ-sobu činnosti v grafickém režimu běží X server i jeho klienti na stejném počítači.

Názvy displejů

Z pohledu uživatele má každý X server svůj *název displeje* ve tvaru:

název_počítače:číslo_displeje.číslo_obrazovky

Aplikace z těchto informací zjistí, jak se má připojit k X serveru a kterou obrazovku použít (na systémech s více monitory).*název_počítače:* Specifikuje počítač, k němuž je displej fyzicky připojen. Pokud název není zadán,použije se nejefektivnější možná metoda komunikace se serverem na lokálním počítači.
číslo_displeje: Termínem „displej“ se obvykle rozumí skupina monitorů, které sdílejí společnou klá-vesnici a ukazovací zařízení (myš, tablet a podobně). Většina pracovních stanic má typicky při-pojenu jen jednu klávesnici, a má tedy jen jeden displej. Větší víceuživatelské systémy často míva-jí připojeno více displejů. Kvůli rozlišení má každý displej své číslo (počínaje nulou), které se při-děluje při startu X serveru daného displeje. Číslo displeje je povinnou složkou názvu displeje.

číslo_obrazovky: Některé displeje sdílí jednu klávesnici a myš mezi dvěma nebo více monitory. Každý monitor zobrazuje svou vlastní sadu oken, proto má každý z nich své číslo obrazovky (počí-naje nulou), čísla se opět přidělují při spuštění X serveru daného displeje. Pokud není v názvu dis-pleje uvedeno číslo obrazovky, předpokládá se obrazovka 0.

Na systémech kompatibilních s normou POSIX je název výchozího displeje uložen v proměnné DISPLAY. Hodnotu této proměnné automaticky nastavuje emulátor terminálu xterm. Pokud se ovšem přihlašujete po síti ke vzdálenému systému, budete možná muset nastavit proměnnou DISPLAY ručně tak, aby se odkazovala na váš displej – viz kapitolu „Systém X Window“, případ-ně to za vás automaticky udělá ssh, jak se dozvíte dále.

Další informace naleznete v manuálových stránkách X (man X).

Správci oken a pracovní plochy

Rozložení oken na obrazovce je řízeno speciálními programy, takzvanými *manažery oken*. Větši-na správců oken se řídí zadanými geometrickými parametry okna, některé je ale mohou ignoro-vat (a například vyžadují, aby uživatel explicitně vyznačil umístění okna na obrazovce myší).

Správce oken je normální klientský program (byť složitý), takže jich existuje celá řada. V distribu-ci konzorcia X se nachází správce twm, většina uživatelů ovšem preferuje propracovanější správ-ce, samozřejmě pokud jim to dovolí výkon systému. Oblíbené jsou například Sawfish a Enligh-tenment, které nabízejí více možností nastavení stylu a vzhledu.

Správce plochy pak využívá služeb toho či onoho správce oken a zajišťuje pohodlné rozvržení pracovní plochy se všemi nabídkovými lištami, vyskakovacími nabídkami, informativními zpráva-mi, hodinami, správci programů, správci souborů a podobně. Mezi nejoblíbenější správce pracovní plochy patří Gnome a KDE, které fungují na prakticky každé linuxové distribuci i na mnoha dal-ších unixových systémech.

> KDE aplikace v Gnome / Gnome aplikace v KDE
>
> K tomu, abyste mohli spouštět aplikace určené pro KDE, nemusíte KDE používat jako správce plochy. Máte-li nainstalovány KDE knihovny (balíček kdelibs), můžete tyto apli kace spouštět prostřednictvím nabídek Gnome nebo přímo z terminálu.
>
> Spouštění aplikací určených pro Gnome v prostředí KDE je poněkud složitější, protože Gnome neobsahuje žádnou základní sadu univerzálních knihoven. Při pokusu o instalaci či spuštění konkrétní aplikace ovšem obvykle snadno přijdete na potřebné závislosti a příslušné balíčky můžete doinstalovat.

Konfigurace X serveru

V Linuxu se obvykle používá distribuce X serveru označovaná *XFree86*. Nastavení serveru je ulo-ženo v souboru XF86Config. Tento soubor mimo jiné nastavuje parametry grafické karty a může být uložen na více místech, typicky se ale nachází v adresáři /etc/X11.

Pokud ve svém systému najdete soubor /etc/X11/XF86Config, další podrobnosti se můžete dočíst v manuálových a informačních stránkách XF86Config.
Vzhledem k licenčním problémům s distribucí XFree86 používají novější systémy distribuci *X.Org*. Hlavní konfigurační soubor se jmenuje xorg.conf a najdete jej opět většinou v adresáři /etc/X11. Soubor se skládá z řady sekcí, které mohou být uvedeny v různém pořadí. Tyto sekce obsahují informace o monitoru, videokartě, nastavení obrazovky, klávesnici a podobně. Jako normální uži-vatel se o obsah tohoto souboru nemusíte příliš starat, protože všechno bývá většinou správně nastaveno při instalaci systému.

Dále v textu se většinou odkazujeme na soubor XF86Config, ale když jej ve svém systému nena-jdete, použijte automaticky xorg.conf – vaše distribuce již s největší pravděpodobností používá X.org. Soubor xorg.conf je podobný původnímu XF86Config, detaily či změny hledejte v man xorg.conf. Další informace o nastavení ovladačů videokaret najdete v přímo manuálových strán-kách se jménem ovladače, např. man radeon, man nv, man i810, viz informace v manuálových stránkách xorg.conf.

Pokud byste potřebovali změnit nastavení grafického serveru, můžete buď použít příslušný konfi-gurační nástroj nebo můžete ručně modifikovat příslušný konfigurační soubor. Více informací naleznete na manuálových stránkách, různé distribuce používají různé vlastní konfigurační nástro-je. Pokud uděláte v konfiguraci grafického systému chybu, nemusí se vám vůbec podařit spustit grafický režim nebo bude nepoužitelný. Proto je rozumné před zahájením změn pořídit kopii stá-vající konfigurace.

Národní nastavení

Nastavení klávesnice

Pro textovou konzolu se rozložení klávesnice nastavuje příkazem loadkeys. Rozložení klávesni-ce pro grafický režim můžete změnit buď odpovídajícím konfiguračním nástrojem vaší distribuce nebo můžete přímo změnit sekci *Keyboard* souboru XF86Config. Konkrétně vás bude zajímat nastavení parametru XkbdLayout:

XkbLayout "us"

Takto může vypadat výchozí nastavení. Můžete je změnit tak, že hodnotu v uvozovkách nahradí-te jinou hodnotou z těch, které najdete v podadresářích adresáře keymaps. Pokud nevíte, kde adresář hledat, použijte příkaz locate:

locate keymaps

Rozložení klávesnice je možné kombinovat, například:

Xkblayout "us,cz"

Než začnete soubor /etc/X11/XF86Config měnit, zálohujte si jej! Budete k tomu potřebovat pra-covat jako uživatel *root*. Po změně nastavení se odhlaste a znovu přihlaste, aby došlo k novému spuštění X serveru.

V prostředí Gnome najdete aplet, kterým se můžete snadno přepínat mezi více rozloženími klá-vesnice, k používání tohoto programu už nepotřebujete žádná speciální práva. Podobný nástroj se nachází i v KDE. Nakonec ještě malá poznámka – pro novější X.org by příslušný řádek s nasta-vením klávesnice vypadal takto:

Option "XkbLayout" "us,cz"

Fonty

Nástrojem setfont nahrajete fonty v textovém režimu. Většina systémů standardně obsahuje sou-bor inputrc, který umožňuje kombinací kláves vytvářet znaky s diakritikou. Správce systému by měl do souboru /etc/bashrc přidat následující řádek:

export INPUTRC=”/etc/inputrc”

Časové pásmo

Informace o časovém pásmu a nastavení času se typicky zadávají při instalaci systému. Následně se přesný čas může udržovat pomocí klienta služby *NTP* (Network Time Protocol). Většina linu-xových distribucí standardně spouští démona ntpd:

debby:~> ps -ef | grep ntpd ntp 24678 1 0 2002 ? 00:00:33 ntpd -U ntp

Přečtěte si manuál ke svému systému a dokumentaci k balíku NTP. Většina správců pracovní plo-chy obsahuje nástroje pro nastavení systémového času, k jejich spuštění potřebujete práva admi-nistrátora systému.

Ke správnému nastavení časového pásma slouží příkazy tzconfig nebo timezone. Časové pásmo se obvykle nastavuje při instalaci. Většina distribucí k tomu používá své vlastní nástroje, jejichž popis naleznete v dokumentaci.

Jazyky

Pokud preferujete výpis systémových zpráv v jiném jazyce než v angličtině, nastavte proměnné prostředí LANG a LANGUAGE, čímž zapnete podporu národních nastavení daného jazyka a případně specifické fonty daného jazyka.

Ve většině grafických přihlašovacích programů, jako jsou například gdm nebo kdm, máte mož-nost zvolit jazyk ještě před přihlášením. Většina dnešních systémů používá výchozí nastavení *en_US.UTF-8, pro češtinu to může být cs_CZ.UTF-8 či ve starších systémech cs_CZ.ISO-8859-2*. Neměl by to být problém, protože progra-my, obsažené v těchto systémech, dokážou s kódováním UTF-8 správně pracovat, takže editorem vi budete schopni editovat jakékoliv soubory, příkaz cat nebude vypisovat nesmysly a podobně.

Problémy mohou nastat, pokud se připojujete ke staršímu systému, který toto kódování nepod-poruje, nebo pokud budete chtít editovat soubor v kódování UTF-8 na systému, který umí pouze jednobajtová kódování. Užitečným nástrojem může být v těchto případech příkaz recode, který umí převést soubory z jednoho kódování do druhého. Podrobnější informace o jeho možnostech a použití naleznete v manuálových stránkách. Druhá možnost je dočasně změnit používané systé-mové kódování prostřednictvím proměnné LANG:

debby:~> acroread /var/tmp/51434s.pdf Warning: charset "UTF-8" not supported, using "ISO8859-1". Aborted

debby:~> set | grep UTF LANG=en_US.UTF-8

debby:~> export LANG=en_US

debby:~> acroread /var/tmp/51434s.pdf <--otevře se nové okno-->

Specifická národní nastavení

Seznam dokumentů HOWTO (*http://www.tldp.org/HOWTO/HOWTO-INDEX/howtos.html*) obsahu-je dokumenty s lokalizačními postupy pro celou řadu jazyků (například běloruštinu, čínštinu, fin-štinu, hebrejštinu, polštinu, slovenštinu a další).

Dokument pro lokalizaci do češtiny bohužel chybí, nicméně většina moderních distribucí jako openSUSE, Mandriva Linux či Fedora Core umí vše výše uvedené nastavit správně podle jazyka vybraného již při instalaci. Úpravy xorg.conf, instalace fontů nebo nastavení proměnných LANG* tak s největší pravděpodobností vůbec nebude nutné.

Instalace nových programů

Úvod

Většina uživatelů bývá příjemně překvapena, že po nainstalování linuxové distribuce má k dispo-zici použitelný počítač – většina distribucí obsahuje podporu celé řady grafických karet, síťových karet, monitorů a dalších zařízení, takže nebývá nutné instalovat speciální ovladače zařízení. Hlav-ní distribuce obsahují rovněž nástroje, jako je kancelářský balík, webové prohlížeče, poštovní kli-enty a další. I tak je ovšem pravděpodobné, že systém po instalaci nebude splňovat všechny vaše požadavky.

Pokud vám nějaký program chybí, je možné, že je sice součástí distribuce, nebyl však nainstalován. Může se také stát, že požadovaný program se v systému nachází, nedělá ale to, co byste potřebovali. Nezapomínejte, že Linux se velmi rychle vyvíjí a některé programy se denně mění. Neztrácejte proto čas řešením problémů, které už jsou možná dávno vyřešeny.

Kdykoliv budete chtít, můžete systém aktualizovat nebo doinstalovat další programy. Většina pro-gramů se dodává v podobě balíčků. Najdete je jednak na instalačních CD, jednak na Internetu. Vhodným místem, kde můžete začít hledat další programy a kde naleznete postup instalace pro-gramů ve vaší distribuci, jsou samozřejmě webové stránky distribuce, viz příloha A. U každého nového programu si vždy přečtěte dokumentaci a případně návod k instalaci. Všechny programy obsahují soubor README, který byste rozhodně vždy měli přečíst.

Formáty balíčků Balíčky RPM

RPM, RedHat Package Manager, je výkonný správce balíčků, jehož prostřednictvím můžete instalovat, aktualizovat a mazat balíčky. Umožňuje vám hledat balíčky a dokáže sdělit, které soubory jsou součástí kterého balíčku. Umožňuje rovněž ověřit autenticitu balíčků stažených z Internetu. Pokročilejší uživatelé mohou vytvářet i vlastní RPM balíčky.

Balíček obsahuje archiv souborů a metadata, která slouží k instalaci a mazání souborů v archivu. Tato metadata zahrnují pomocné skripty, atributy souborů a popisné informace o balíčku. Balíč-ky existují ve dvou variantách: binární balíčky, sloužící k „zabalení“ programů pro instalaci, a zdro-jové balíčky, které obsahují zdrojový kód a postup potřebný pro vytvoření binárního balíčku.

Balíčky RPM podporuje celá řada distribucí, například RedHat Enterprise Linux, Mandriva (dříve Mandrake), Fedora Core a openSUSE (dříve SUSE Linux). Kromě dokumentace vaší konkrétní distribuce byste neměli zapomenout ani na příkaz man rpm.

Většinu balíčků nainstalujete jednoduše volbou pro aktualizaci (-U jako *upgrade*), bez ohledu na to, zda balíček již v systému existuje nebo ne. Balíček obsahuje aktuální verzi programu, která se buď přímo nainstaluje nebo přepíše případnou existující starší verzi. Typický příklad vypadá takto:

rpm -Uvh /cesta/k/balíčku(ům)

Přepínač -v zapíná podrobnější výpis, přepínač -h zobrazuje ukazatel průběhu:

[root@jupiter tmp]# rpm -Uvh totem-0.99.5-1.fr.i386.rpmPreparing... ### [100%]
1:totem ### [100%] [root@jupiter tmp]#

Balíčky s novým jádrem se nicméně instalují volbou -i, takže nedojde k přepsání stávajících balíč-ků. Díky tomu budete schopni nabootovat systém se starším jádrem v případě, že by nové nefun-govalo.

Příkazem rpm můžete také ověřit, zda je v systému určitý balíček nainstalován:

[david@jupiter ~] rpm -qa | grep vim vim-minimal-6.1-29 vim-X11-6.1-29 vim-enhanced-6.1-29 vim-common-6.1-29

Můžete také zjistit, který balíček obsahuje určitý soubor:

[david@jupiter ~] rpm -qf /etc/profile setup-2.5.25-1

[david@jupiter ~] which cat cat is /bin/cat

[david@jupiter ~] rpm -qf /bin/cat coreutils-4.5.3-19

K tomu, abyste příkazem rpm mohli prohlížet databázi nainstalovaných balíčků, nepotřebujete práva administrátora. Jako *root* musíte pracovat pouze při přidávání, modifikaci nebo mazání balíčků. Poslední příklad ukazuje, jak příkazem rpm balíček odinstalovat:

[root@jupiter root]# rpm -e totem [root@jupiter root]#

Všimněte si, že odinstalování standardně proběhne úplně potichu. Pokud jste nedůvěřiví, můžete

následně příkazem rpm -qa ověřit, že balíček byl opravdu odinstalován.RPM toho samozřejmě dokáže mnohem více než jen to, co jsme si zde předvedli. Mnoho dalšíchinformací naleznete v dokumentu RPM HOWTO, *http://www.tldp.org/HOWTO/RPM-HOWTO/index.html.*

Balíčky DEB (.deb)

Tento formát balíčků standardně používá Debian GNU/Linux, kde jako nástroje pro správu balíč-ků slouží programy dselect a nověji aptitude. Slouží k výběru balíčků, které chcete instalovat nebo aktualizovat, používá se také při samotné instalaci systému a umožňuje vám definovat pou-žívané přístupové metody, vypsat dostupné balíčky a konfigurovat balíčky.

Všechny potřebné informace naleznete na webových stránkách distribuce Debian, *http://www.us.debian.org/,* kde je mimo jiné dokument „dselect Documentation for Beginners“.
Popularita formátu DEB se stále rozšiřuje, momentálně jej používá pět z deseti nejrozšířenějších distribucí. I na systémech, které nepoužívají balíčky DEB, se navíc stále zvětšuje obliba nástroje apt-get, viz kapitolu „APT“.

Zdrojové balíčky

Velká většina linuxových programů spadá do kategorie Open Source, proto jsou k dispozici v podobě zdrojových balíčků. Zdrojový balíček obsahuje zdrojové kódy potřebné k sestavení vlastní verze programu. Zdrojové kódy můžete typicky nalézt na webových stránkách příslušného programu, nejčastěji v podobě archivu *tgz* (program-verze.tar.gz nebo něco podobného). Pro distribuce založené na RPM často existují balíčky program-verze.src.rpm. Debian a odvozené distribuce nabízejí upravené zdrojové kódy, které můžete stáhnout příkazem apt-get.

Součástí zdrojového balíčku bývá soubor README, ve kterém jsou popsány požadavky, závislos-ti a pokyny pro instalaci. Ve většině případů budete potřebovat překladač jazyka C, gcc. Tento GNU překladač je součástí většiny linuxových systémů a byl přenesen i na celou řadu jiných plat-forem.

Automatická správa a aktualizace balíčků Obecné poznámky

Jako první krok po instalaci nového systému byste měli provést jeho aktualizaci. Platí to pro všech
ny operační systémy a Linux není výjimkou.
Aktualizace většiny linuxových systémů typicky naleznete na nejbližším zrcadle používané distri
buce. Seznamy zrcadel bývají uvedeny přímo na stránkách distribuce, viz kapitolu „Jak dále“.

Systém byste měli aktualizovat pravidelně, nejlépe denně – i jednou za několik týdnů je ale lepší než vůbec. Vždy byste se měli snažit mít co nejaktuálnější verzi distribuce, protože Linux se trva-le vyvíjí. Jak už bylo řečeno, neustále se objevují nové funkce, vylepšení a opravy chyb, čas od času dochází i k odstranění závažných bezpečnostních problémů.

Příjemné je, že většina linuxových distribucí poskytuje nástroje, díky nimž nemusíte provádět aktualizace ručně. O těchto nástrojích budeme hovořit v následující části. Celé téma je samozřejmě mnohem rozsáhlejší, například i u zdrojových balíčků lze zajistit automatickou aktualizaci. My se budeme zabývat pouze nejčastějšími případy. Doporučené postupy naleznete v dokumentaci své distribuce.

APT

Advance Package Tool je systém pro správu softwarových balíčků. Řádkový nástroj pro správu balíčků se jmenuje apt-get a jeho součástí je vynikající manuálová stránka, která popisuje, jak balíčky instalovat a aktualizovat a jak aktualizovat nejen jednotlivé balíčky, ale i celou distribuci. APT pochází z distribuce Debian GNU/Linux, kde slouží jako výchozí správce balíčků. Kromě toho byl APT portován i pro práci s balíčky RPM. Hlavní výhodou APT je to, že je k dispozici zdarma a je velmi flexibilní. Lze jej nastavit pro potřeby různých distribucí a dokáže se chovat i jako někte-ré komerční systémy pro správu balíčků.

Obecně platí, že při prvním použití apt-get musíte vytvořit seznam dostupných balíčků. Provede-te to příkazem:

apt-get update

Následně systém aktualizujete takto:

apt-get upgrade

Aktualizujte často, jde o snadný způsob, jak mít systém aktuální a bezpečný.Kromě tohoto obecného použití dokáže apt-get také velmi rychle instalovat nové balíčky. Napří-klad:

```
[david@jupiter ~] su - -c "apt-get install xsnow"Password:Reading Package Lists... DoneBuilding Dependency Tree... DoneThe following NEW packages will be installed:
    xsnow 0 packages upgraded, 1 newly installed, 0 removed and 3 not upgraded. Need to get 33.6kB of archives. After unpacking 104kB of additional disk space will be used. Get:1 http://ayo.freshrpms.net redhat/9/i386/os xsnow 1.42-10 [33.6kB] Fetched 33.6kB in 0s (106kB/s) Executing RPM (-Uvh)... Preparing... ########################################### [100%]
                                    1:xsnow ########################################### [100%]
```

Všimněte si volby -c u příkazu su, ta říká, že se má pod superuživatelským účtem provést pouzenásledující příkaz a dál se má pracovat opět v prostředí původního uživatele. Tak se vám nesta-ne, že byste se zapomněli ze superuživatelského účtu odhlásit. Pokud instalovaný balíček závisí na dalších balíčcích, apt-get stáhne i nainstaluje všechny potřeb-né balíčky.Další informace naleznete v APT HOWTO, *http://www.debian.org/doc/user-manuals#apt-howto.*

Systémy založené na RPM balíčcích

Program Update Agent, který původně podporoval jen RPM balíčky systému RedHat, existuje dnes v rozšířené podobě, která podporuje i jiné zdroje než RedHat. Tento nástroj představuje úplný systém pro aktualizaci RPM balíčků na systémech RedHat nebo Fedora Core. Z příkazového řádku provedete aktualizaci systému zadáním příkazu up2date. Kromě toho je ve standardní grafické instalaci na ploše zobrazena ikona, která indikuje, zda jsou dostupné nové aktualizace.

V poslední době získává stále větší popularitu nástroj Yellowdog's Updater Modified (yum). Jde o interaktivní automatický program pro instalaci, aktualizaci a mazání nainstalovaných RPM balíč-ků. Na systémech Fedora Core jde o standardně preferovaný nástroj. Počínaje verzí SuSE Linux 7.1 je možno aktualizace provádět i prostřednictvím webového rozhraní a nástroje YOU, Yast Online Update.

Mandriva Linux nabízí nástroje URPMI, což je sada wrapperů, které usnadňují instalaci nových pro-gramů. V kombinaci s nástroji RPMDrake a Mandriva Update nabízejí všechno potřebné pro snad-nou instalaci a odinstalaci softwarových balíčků. Mandriva Online nabízí celou řadu služeb a doká-že administrátora upozornit, pokud jsou k dispozici aktualizace pro konkrétní systém. Více informací získáte mimo jiné příkazem man urpmi (manuálová stránka urpmi je k dispozici i v češtině).

Rovněž grafická prostředí KDE a Gnome nabízejí vlastní grafické verze správců balíčků.

Aktualizace jádra

Většina instalací Linuxu bude bezproblémově fungovat, pokud distribuci pravidelně aktualizujete. Při aktualizaci může podle potřeby dojít k instalaci nového jádra a provedení všech potřebných změn. Ruční překlad a instalaci jádra byste měli provádět pouze v případě, že vyžadujete funkce, které distribuční jádro standardně neobsahuje.
Ať už překládáte vlastní optimalizované jádro anebo použijete předpřipravený balíček, nainstalujte je vždy v koexistenci se stávajícím jádrem, abyste měli jistotu, že systém bude funkční i v případě, že nové jádro nebude pracovat správně.

Následně upravte konfigurační soubor grub.conf zavaděče tak, abyste při bootování mohli zvolit obě jádra. Jednoduchý příklad může vypadat takto:

```
# grub.conf generated by anaconda # # Note that you do not have to rerun grub after making config changes.

#               You have a /boot partition.             This means that
NOTICE:
#               all kernel and initrd paths are relative to /boot/, e.g.
#               root (hd0,0)
#               kernel /vmlinuz-version ro root=/dev/hde8
#               initrd /initrd-version.img

#boot=/dev/hde default=0 timeout=10 splashimage=(hd0,0)/grub/splash.xpm.gz title Red Hat Linux new (2.4.9-31)
root (hd0,0) kernel /vmlinuz-2.4.9-31 ro root=/dev/hde8 initrd /initrd-2.4.9-31.img
title old-kernel
root (hd0,0)
kernel /vmlinuz-2.4.9-21 ro root=/dev/hde8
initrd /initrd-2.4.9-21.img
```

Až si ověříte funkčnost nového jádra, můžete z konfiguračního souboru odstranit řádky, které umožňují nabootování staršího jádra. Doporučuje se s tímto krokem pár dní vyčkat, než se nové jádro dostatečně osvědčí.

Instalace dodatečných balíčků z instalačních CD Připojení CD

Postup instalace se nijak neliší od jiných způsobů ruční instalace balíčků, rozdíl je pouze v tom, že CD musíte nejprve připojit do souborového systému. Na většině systémů k tomu dojde auto-maticky po vložení disku do mechaniky, protože při startu systému se ve standardním nastavení spouští démon automount. Pokud se CD automaticky nepřipojí, spusťte v okně terminálu příkaz mount. V závislosti na konkrétní konfiguraci systému většinou poslouží něco takového:

mount /dev/cdrom /mnt/cdrom

Na některých systémech může vyjímatelná média připojovat pouze *root*, to záleží na nastavení. Kvůli usnadnění připojení mívá CD mechanika většinou záznam v souboru /etc/fstab, který obsa-huje seznam souborových systémů a jejich připojovacích bodů tak, jak vytvářejí celý strom sou-borového systému. Příslušný řádek může vypadat například takto:

[david@jupiter ~] grep cdrom /etc/fstab /dev/cdrom /mnt/cdrom iso9660 noauto,owner,ro 0 0

Takto nastavený systém bude rozumět příkazu mount /mnt/cdrom. Volba noauto znamená, že k připojení CD mechaniky nedochází v době startu systému.
Můžete také zkusit pravé tlačítko myši na ikoně CD na pracovní ploše, souborový manažer možná zvládne připojení mechaniky sám. Zda je mechanika připojena, můžete zjistit příkazem mount bez parametrů:

[david@jupiter ~] mount | grep cdrom/dev/cdrom on /mnt/cdrom type iso9660 (ro,nosuid,nodev)

Použití CD

Po připojení CD mechaniky můžete přejít do adresáře odpovídajícího bodu připojení, obvykle tedy /mnt/cdrom, kde uvidíte přímo obsah CD disku. Pracovat s nimi můžete naprosto stejnými příkazy a postupy, jaké se používají pro soubory na pevném disku.

Vysunutí CD

Abyste mohli CD disk vyjmout z mechaniky, nesmí se souborový systém na tomto disku použí-vat. Jako „používaní souborového systému“ se chápe i to, že se nacházíte v některém podadresá-ři přípojného bodu, tedy v našem případě adresáře /mnt/cdrom. Nejprve tedy musíte tyto adresá-ře opustit. Můžete to provést například zadáním příkazu cd bez parametrů, kterým se přesunete do svého domovského adresáře. Následně můžete použít buď příkaz:

umount /mnt/cdrom

nebo:

eject *cdrom*

> Zablokovaná mechanika
>
> Nikdy nepostupujte násilně. Trik s kancelářskou sponkou není rozumný, protože i pokud se vám podaří disk vyjmout, systém si bude i nadále myslet, že je disk přítomen, jelikož neproběhlo požadované odpojení. Může se dokonce stát, že budete v takovém případě nuceni celý systém rebootovat, abyste jej dostali zpět do konzistentního stavu.
>
> Pokud stále dostáváte chybové hlášení „device busy“, zkontrolujte, že se v žádné z relací shellu nenacházíte v některém z adresářů na CD ani že nemáte žádný z adresářů otevřený v grafickém prostředí. Pokud se vám nedaří zjistit, kdo souborový systém používá, zkuste příslušný proces vystopovat příkazem lsof.

Shrnutí

Pokud má všechno své místo, je tím vyřešena polovina problémů.
Kromě udržení pořádku je rovněž důležité to, abyste se ve svém domovském prostředí, ať už tex tovém nebo grafickém, cítili příjemně. Textové prostředí se nastavuje prostřednictvím konfigurač ních souborů shellu. Vlastnosti grafického prostředí primárně závisí na konfiguraci X serveru, jehož služeb pak využívá celá řada dalších aplikací, jako jsou správci oken a pracovní plochy.

I tyto programy mají své konfigurační soubory. Podrobnosti o jejich konfiguraci naleznete v doku-mentaci k systému a ke konkrétním programům.Lokalizační nastavení, jako jsou nastavení klávesnice, fontů a jazykové podpory, se nejlépe reali-zují při instalaci.
Nainstalované programy se spravují buď automaticky nebo ručně prostřednictvím balíčkovacího systému distribuce.

Cvičení

Prostředí shellu

Vytiskněte si nastavení prostředí. Ve které proměnné může být uložen typ procesoru?
Vytvořte skript, který něco vypíše. Nastavte mu práva tak, aby mohl být spuštěn. Otestujte skript.
Ve svém domovském adresáři vytvořte podadresář a zkopírujte skript do něj. Přidejte adre-sář trvale do své prohledávací cesty.
Ověřte, že skript lze spustit bez explicitního zadání jeho umístění.
Vytvořte v domovském adresáři podadresáře pro uložení různých druhů dat, například adresář music pro uložení hudby nebo documents pro uložení poznámek a podobně. Používejte je!
Vytvořte si vlastní prompt.
Zobrazte limity systémových prostředků. Můžete je změnit?
Zkuste si přečíst komprimovanou manuálovou stránku, aniž byste ji nejprve dekomprimovali.
Vytvořte alias lll, který provede příkaz ls -la.
Proč příkaz tail testfile > testfile nefunguje?
Připojte datové CD, například distribuční CD Linuxu, a prohlédněte si jeho obsah. Nezapomeňte nakonec disk odpojit.
Skript z kapitoly „Několik jednoduchých příkladů“ není dokonalý. Pro adresáře hlásí chyby. Upravte skript tak, aby kopíroval pouze soubory. K výběru souborů použijte příkaz find. Nezapomeňte skriptu nastavit právo spuštění.

Grafické prostředí

Vyzkoušejte všechna tlačítka myši na různých místech (v terminálu, na pozadí, panelu úloh).
Projděte nabídky příkazů.
Upravte si nastavení terminálového okna.
Pomocí tlačítek myši zkopírujte a vložte text z jednoho terminálového okna do druhého.
Zkuste změnit nastavení správce oken, vyzkoušejte více pracovních ploch (virtuálních obrazovek).
Přidejte na panel úloh nějaký aplet, například monitor zatížení systému.
Nastavte jiné téma grafického prostředí.
Zapněte funkci autofokusu oken – tedy funkci, při níž se okno aktivuje tím, že do něj vstoupíte myší, bez nutnosti na okno klepnout.
Vyzkoušejte jiného správce oken.
Odhlaste se a zvolte jiný typ relace – pokud jste například používali Gnome, vyzkoušejte KDE. Znovu proveďte předchozí kroky.

Tiskárny a tisk

V této kapitole se dozvíme o tiskárnách a o tisknutí souborů. Po přečtení této kapitoly budete schopni:

Formátovat dokumenty
Zobrazit náhled dokumentu před odesláním k tisku
Zvolit vhodnou tiskárnu, která bude s linuxovým systémem spolupracovat
Tisknout soubory a ověřit stav tiskárny
Řešit problémy s tiskem
Nalézt dokumentaci nezbytnou k instalaci tiskárny

Tisk souborů

Obecné

RedHat obsahuje LPRng, vylepšenou verzi známého unixového tiskového systému. Jakmile tis-kárnu jednou nastavíte, bude vám stačit už jen naučit se používat příkaz lpr, kterým se soubory posílají na tiskárnu. V zásadě jde pouze o příkaz:

```
lpr file(s)
```

Příkaz lpr spolupracuje s tiskovým démonem lpd, který požadovaný soubor vytiskne v okamži-ku, kdy je tiskárna k dispozici. Pokud nezadáte název souboru, čtou se data ze standardního vstu-pu (například v situaci, kdy příkazu lpr pomocí roury předáte výstup jiného příkazu). Příkaz lpr má celou řadu voleb, můžete je vypsat pomocí volby --help. Podrobný popis naleznete na infor-mačních stránkách. Kromě toho byste ve svém systému měli najít i příkaz lp, kvůli kompatibilitě s unixovými systémy. Snadno zjistíte, že lp je pouze symbolický odkaz na lpr:

davy:~> ls -l /usr/bin/lp* lrwxrwxrwx 1 root root 3 Oct 28 14:21 /usr/bin/lp -> lpr -rwxr-xr-x 1 lp lp 395192 Aug 11 2001 /usr/bin/lpq -rwxr-xr-x 1 lp lp 408536 Aug 11 2001 /usr/bin/lpr -rwxr-xr-x 1 lp lp 392984 Aug 11 2001 /usr/bin/lprm -rwxr-xr-x 1 root root 4651 Oct 19 22:17 /usr/bin/lprsetup.sh -rwxr-xr-x 1 lp lp 398488 Aug 11 2001 /usr/bin/lpstat davy:~> ps -ef | grep lpd lp 1003 1 0 Feb22 ? 00:00:00 lpd Waiting

Jakmile je soubor zařazen do tiskové fronty, je mu přiřazeno číslo tiskové úlohy:

davy:~> lp /etc/profile request id is davy@blob+253
Obsah tiskové fronty můžete vypsat příkazem lpq. Zadáte-li jej bez parametrů, vypíší se položky ve výchozí tiskové frontě.

davy:~> lpq
Printer: lp@blob
  Queue: no printable jobs in queue
  Status: job 'cfA284blob.somewhere.org' removed at 11:02:47.098

Pokud chcete některou z položek ve frontě odstranit, použijte příkaz lprm. Zadáte-li lprm -,vymažou se všechny úlohy ve frontě. Chcete-li zrušit jen jednu určitou úlohu, zadejte jako para-metr příkazu lprm číslo této úlohy.

V rozsáhlejších prostředích je možné příkazem lpc ovládat více tiskáren. Podrobnosti k jednotli-vým příkazům naleznete na informačních stránkách.Existuje celá řada grafických tiskových nástrojů, které fungují jako front-end příkazu lpr. Většinagrafických aplikací k tisku interně používá příkaz lpr. Podrobnosti k jednotlivým programůmnaleznete v jejich nápovědě a dokumentaci.

Uživatelé distribucí s novějším tiskovým systémem CUPS (Mandriva Linux, openSUSE) mohoupoužívat stejné příkazy, jako jsme uvedli výše. CUPS obsahuje vlastní varianty těchto příkazů,a pokud byste potřebovali, obsahuje také tiskového démona kompatibilního s lpd (hledejte cups-lpd). Nápovědu pro CUPS najdete v manuálových stránkách jednotlivých příkazů (man lpra další), případně – běží-li CUPS na lokálním počítači – na adrese *http://localhost:631/help.*

Formátování Nástroje

Chceme-li dostat z tiskárny něco rozumného, musíme soubory nejprve naformátovat. Bez ohledu na celou řadu existujících formátovacích programů v Linuxu téměř vždy najdete základní unixové formátovací nástroje a jazyky.

Pro větší nebo opakované úkony (například při tisku výstupu ze skriptů) jsou vhodnější značkovací jazyky, kdy formátování dokumentu řídí přímo počítač.

groff: GNU verze unixového příkazu roff. Jde o front-end k formátovacímu systému groff. Ve většině příkazů spustí příkaz troff a postprocesor odpovídající zvolenému zařízení. Umožňuje generovat soubory ve formátu PostScript.

■ *TeX* a balík maker *LaTeX*: jeden z nejčastěji používaných značkovacích jazyků na unixo-vých systémech. Obvykle se spouští příkazem tex, který naformátuje soubor a vytvoří odpovídající reprezentaci sázeného textu nezávislou na výstupním zařízení.

Technické dokumenty se *stále* velmi často sází v LaTeXu, protože podporuje sazbu mate-matických vzorců, a to bez ohledu na snahy W3C (*http://www.w3.org,* World Wide Web Consortium) zahrnout tyto funkce do jiných aplikací.

SGML a XML: Pro Unix a Linux existují volně dostupné parsery těchto jazyků. XML je další generací jazyka SGML, představuje základ dokumentačního formátu DocBook XML.

Linux obsahuje celou řadu formátovacích nástrojů, například pdf2ps, fax2ps nebo a2ps. Kromě těchto řádkově orientovaných nástrojů existuje celá řada grafických textových editorů. K dispozici je několik kompletních kancelářských balíků, řada z nich je dostupných zdarma. Jme-nujme jenom některé: AbiWord, KWord, OpenOffice.org, Applix, WordPerfect a další.

Tisk dokumentace

Manuálové stránky jsou uloženy jako předformátovaná data pro příkaz troff, před tiskem je nutné je naformátovat. Tisk se provede volbou -t:

man -t příkaz > man-příkaz.ps

Vzniklý postscriptový soubor můžete vytisknout. Pokud máte nakonfigurovánu výchozí tis kárnu, stačí zadat jen man -t příkaz a stránka se rovnou vytiskne.

Chcete-li tisknout informační stránky, přečtěte si informační stránku příkazu info.

Náhled souborů před tiskem

Co můžete poslat na tiskárnu, to můžete také zobrazit na obrazovce. V závislosti na formátu sou-borů můžete použít některý z následujících příkazů:

Pro soubory ve formátu PostScript použijte příkaz gv (GhostView).
Pro *dvi* soubory systému TeX použijte příkaz xdvi.
Pro soubory PDF použijte prohlížeč xpdf, gv nebo přímo acroread od Adobe, který je rovněž k dispozici zdarma.

Strana serveru

Obecné

Ještě před několika lety byla volba tiskového systému pro Linux velmi jednoduchá – všichni pou
žívali stejný starý LPD, převzatý téměř úplně z BSD Net-2. V současné době už je k dispozici více tiskových systémů. Některé distribuce obsahují LPRng, pře-pracovanou verzi původní Line Printer Daemona (LPD) z BSD Unixu. LPD je rovněž název síťového tiskového protokolu. Tento protokol používá nejen démon LPD, ale v zásadě všechny síťo-vé tiskové servery, síťové tiskárny a všechny známé tiskové systémy. LPD představuje základní úroveň kompatibility v síťovém tisku, protože síťový tisk protokolem LPD lze snadno nastavit i v systémech Microsoft Windows, které tak mohou jednoduše tisknout na tiskárně připojené k linuxovému stroji.

Systém LPRng je mnohem lepší implementací standardního LPD. Pokud tedy potřebujete služeb LPD používat, zvolte raději implementaci LPRng. K nastavení do žádoucího stavu je zapotřebí mnohem méně zaklínání a několik zbývajících zaklínacích pasáží je velmi dobře dokumentováno.

LPRng se mnohem snáze administruje v případě větších instalací (čti: více než jedna tiskárna, jaká-koliv sériová tiskárna, jakákoliv síťová tiskárna nepodporující LPD) a obsahuje mnohem méně potenciálně rizikového kódu než standardní implementace. Dokonce o sobě může prohlašovat, že je bezpečný – neobsahuje žádné SUID binární soubory a podporuje autentizaci přes PGP nebo Kerberos.

Zajímavým novým projektem je CUPS, Common UNIX Print System, implementace protokolu IPP (Internet Printing Protocol), což je protokolu HTTP podobný RFC standard, nahrazující starý a nemotorný protokol LPD. CUPS je distribuován pod licencí GNU Public License a je to výchozí tiskový systém v MacOSa některých distribucích (Mandriva Linux).

Více informací naleznete na stránkách *http://www.cups.org*.

Grafická konfigurace tiskárny

Většina distribucí obsahuje grafický nástroj pro konfiguraci síťových i místních tiskáren. Umožňu-jí vám zvolit typ tiskárny ze seznamu a snadno otestovat její funkčnost. Nemusíte se tak starat
o syntaxi a umístění konfiguračních souborů. Než tedy začnete instalovat tiskárnu, nahlédněte do dokumentace ke své distribuci.

Koupě tiskárny pro Linux

Linuxové jádro vám umožňuje komunikovat s jakoukoliv tiskárnou, kterou připojíte k sériovému, paralelnímu či USB portu, plus s jakoukoliv tiskárnou připojenou k síti. To samo o sobě ale nestačí, musíte být schopni rovněž vygenerovat data ve formátu, kterému tiskárna rozumí.

Linux podporuje prakticky všechny tiskárny kompatibilní s modely HP a/nebo IBM. Obecně platí,
že jakékoliv tiskárny použitelné v Unixu budou fungovat i v Linuxu. Tiskárny, které obsahují pouze ovladače pro Win9x či WinXP, mohou být problematické, není-li pro ně k dispozici žádná další podpora. Váháte-li, nahlédněte do HOWTO dokumentu týkajícího se hardwarové kompatibility. Kompletní seznam kompatibilních zařízení včetně detailů o stavu a kvalitě jejich podpory v Linuxu najdete na *http://www.linuxprinting.org*.

Nejlepší volbou je tiskárna, jejíž firmware nativně podporuje PostScript, protože prakticky všechny unixové a linuxové programy, které produkují tiskový výstup, jej dokáží produkovat v Post-Scriptu, což je průmyslový standard v oblasti tisku. Postscriptové tiskárny jsou obvykle o něco dražší, získáváte však podporu na zařízení nezávislého, otevřeného jazyka.

Problémy s tiskem

V této části budeme hovořit o tom, co můžete jako uživatel udělat, pokud něco přestane fungo-vat. Nebudeme se zabývat problémy, které se týkají samotného tiskového démona, protože jeho nastavení je úkolem správce systému.

Špatné soubory

Pokud zadáte k tisku špatný soubor, můžete jeho tisk zrušit příkazem lprm číslo_úlohy, kde číslo úlohy uvádíte ve tvaru *název_tiskárny-číslo*, což jsou údaje, které zjistíte z výpisu příkazu lpq. Tento zásah bude fungovat v případě, že úloha stále čeká v tiskové frontě. Pokud jste jediným uživatelem tiskárny, musíte být hodně rychlí, protože úlohy se typicky odesílají na tiskárnu po několika sekundách. Jakmile se jednou úloha začne tisknout, standardními linuxovými nástroji už s ní nic nenaděláte.

Tiskárna netiskne

Pomocí příkazu lpq zjistěte, zda tiskovou úlohu najdete ve frontě:

```
elly:~> lpq
Printer: lp@blob
Queue: 2 printable jobs
Server: pid 29998 active
Unspooler: pid 29999 active
Status: waiting for subserver to exit at 09:43:20.699

Rank Owner/ID                 Class Job Files                 Size Time
1    elly@blob+997            A     997 (STDIN)               129 09:42:54
2    elly@blob+22             A     22 /etc/profile           917 09:43:20
```

Řada tiskáren má dnes webové rozhraní, takže do prohlížeče stačí zadat IP adresu tiskárny a zobrazí se informace o stavu tiskárny.

Pokud úlohu nenajdete ani ve frontě, ani na tiskárně, obraťte se na správce systému. Pokud se úloha nachází ve frontě, ověřte, zda tiskárna právě netiskne něco jiného. Pokud ano, pak prostě počkejte, vaše úloha přijde na řadu.

Jestliže tiskárna netiskne, zkontrolujte, zda je v ní papír a zda je fyzicky zapojena jak do napáje-ní, tak k počítači či síti. Je-li všechno v pořádku, možná pomůže tiskárnu restartovat. Požádejte o radu správce.Jde-li o síťovou tiskárnu, zkuste tisk z jiného počítače. Je-li tiskárna z vašeho počítače dostupná(viz příkaz ping v kapitole „Sítě“), můžete zkusit naformátovaný soubor (například soubor.ps v případě postscriptové tiskárny) poslat na tiskárnu přímo pomocí FTP klienta. Jestliže se soubor vytiskne, máte špatně nastavený počítač. Jestliže nefunguje ani to, tiskárna možná nero-zumí formátu souboru, který na ni posíláte.Další tipy a návody naleznete na stránkách GNU/Linux Printing, *http://www.linuxprinting.org*.

Shrnutí

Tiskové služby v Linuxu jsou obsluhovány sadou nástrojů založených na unixovém standardu LPD. Seznam příkazů ukazuje následující tabulka:

Příkaz	Popis
	Vytiskne soubor.
	Vypíše stav tiskové fronty.
	Odstraní úlohu z fronty.
	Prohlížeč PDF souborů.
	Formátovací nástroj.
	Prohlížeč postscriptových a PDF souborů.
	Nástroj pro nastavení tiskárny.
	Prohlížeč souborů DVI.
	Prohlížeč souborů PDF.
	Zkonvertuje soubor na PostScript.

Příkazy související s tiskem

Cvičení

Ke konfiguraci a testování tisku potřebujete mít k dispozici tiskárnu a práva superuživatele. Pokud obojí máte, můžete vyzkoušet následující:

Prostřednictvím grafického rozhraní svého systému nainstalovat tiskárnu.
Prostřednictvím tohoto rozhraní vytisknout zkušební stránku.
Vytisknout nějakou stránku příkazem lp.
Tisk z nějaké aplikace, například z Mozilly nebo z OpenOffice.org, příkazem Soubor -> Tisk.
Odpojit tiskárnu od sítě nebo lokálního počítače či tiskového serveru. Co se stane, budete-li teď chtít tisknout?

Následující cvičení můžete vyzkoušet i bez tiskárny či bez superuživatelských práv:

Vytvořte postscriptový soubor z různých zdrojů (například z HTML souboru, z PDF či z manuálové stránky). Prohlédněte si výsledek příkazem gv.
Ověřte, že běží tiskový démon.
Zkuste vytisknout nějaký soubor. Co se stane?
Pomocí Mozilly vytvořte postscriptový soubor. Prohlédněte si jej příkazem gv.
Převeďte jej do formátu PDF. Prohlédněte si jej příkazem xpdf.

Jak byste z příkazového řádku vytiskli soubor GIF?
Příkazem a2ps převeďte soubor /etc/profile do výstupního souboru. Výsledek opět ověř-te programem gv. Co se stane, neuvedete-li název výstupního souboru?

Základní techniky zálohování

Dříve či později vždy dojde k nějaké nehodě. V této kapitole budeme hovořit o způsobech, jak data bezpečně uložit na jiný počítač, diskety, CD-ROM a pásky. Zmíníme se také o nejoblíbeněj-ších komprimačních a archivačních programech.

Po přečtení této kapitoly budete vědět, jak:

Vytvářet, prohlížet a rozbalit souborové archivy
Pracovat s disketami a jak vytvořit bootovací disketu svého systému
Zapisovat na CD-ROM
Vytvářet inkrementální zálohy
Vytvářet javové archivy
V dokumentaci zjistit, jak se používají další zálohovací zařízení a programy

Úvod

Přestože je Linux jedním z nejbezpečnějších existujících operačních systémů a je navržen pro trva-lý běh, i tak může dojít ke ztrátě dat. Ke ztrátám dat nejčastěji dochází v důsledku chyby uživate-le, může k nim však dojít i při výpadku systému, například v důsledku výpadku napájení. Vždy je proto rozumné mít citlivá a/nebo důležitá data zkopírována ještě na jiném místě.

Příprava dat Archivace programem tar

Ve většině případů nejprve zálohovaná data uložíme do jediného archivu, který následně zkom-primujeme. Při procesu archivace slučujeme všechny potřebné soubory a eliminujeme nevyužíva-ná prázdná místa. V Linuxu se k tomu většinou používá příkaz tar. Příkaz tar byl původně navr-žen k archivaci dat na pásku, dokáže ale také vytvářet archivy, označované jako *tarballs*.

Program tar má mnoho parametrů, mezi nejdůležitější patří tyto:

- -v: „ukecaný“ režim,
- -t: test, zobrazí obsah tarballu,
- -x: vybaluje z archivu,
- -c: vytvoří archiv,
- -f archivační_zařízení: zadané zařízení bude použito jako zdroj/cíl tarballu, výchozí hod-nota je první páskové zařízení (obvykle /dev/st0 nebo něco podobného), -j: filtr přes bzip2, viz kapitolu „Komprimace a dekomprimace programy gzip nebo bzip2“.

Při zadávání parametrů programu tar se velmi často vynechává uvozující pomlčka.

> Kvůli kompatibilitě používejte GNU tar Archivy vytvořené proprietární verzí programu tar na jednom systému mohou být nekom-patibilní s programem tar na jiném proprietárním systému. Tím může docházet k velkým nepříjemnostem, například pokud systém, na němž byla záloha vytvořena, už není k dis-pozici. Vyhnete se tomu tak, že budete vždy používat GNU verzi programu tar. Linux vždy používá tuto verzi. Pracujete-li na jiném unixovém systému, příkazem tar --help můžete zjistit, jakou verzi používáte. Nejde-li o GNU verzi, obraťte se na správce systému.

Následující příklad ukazuje vytvoření a rozbalení archivu.

```
gaby:~> ls obrazky/me+tux.jpg nimf.jpg

gaby:~> tar cvf obrazky-s-adresarem.tar images/obrazky/obrazky/nimf.jpgobrazky/me+tux.jpg
```

```
gaby:~> cd obrazky

gaby:~/obrazky> tar cvf obrazky-bez-adresare.tar *.jpg me+tux.jpg nimf.jpg

gaby:~/obrazky> cd

gaby:~> ls */*.tar obrazky/obrazky-bez-adresare.tar

gaby:~> ls *.tar obrazky-s-adresarem.tar

gaby:~> tar xvf obrazky-s-adresarem.tar obrazky/ obrazky/nimf.jpg obrazky/me+tux.jpg

gaby:~> tar tvf obrazky/obrazky-bez-adresare.tar -rw-r--r-- gaby/gaby 42888 1999-06-30 20:52:25 me+tux.jpg
-rw-r--r-- gaby/gaby 7578 2000-01-26 12:58:46 nimf.jpg

gaby:~> tar xvf obrazky/obrazky-bez-adresare.tar me+tux.jpg nimf.jpg

gaby:~> ls *.jpg me+tux.jpg nimf.jpg
```

Příklad rovněž ukazuje rozdíl mezi archivovaným adresářem a archivovanou skupinou souborů. Doporučuje se vždy archivovat celé adresáře, aby při rozbalení archivu nedošlo k promíchání již existujících souborů se soubory vybalenými z archivu.

Pokud máte k počítači připojenu páskovou jednotku a je správně nastavena, můžete namísto názvu souboru s příponou .tar rovnou uvést název páskového zařízení, například:

```
tar cvf /dev/tape mail/
```

Adresář mail a všechny v něm uložené soubory budou tímto příkazem sbaleny do archivu a zapsány na pásku. Protože jsme použili volbu v, budou při archivaci vypisovány názvy archivovaných souborů.

Inkrementální zálohy programem tar

Pomocí volby -N podporuje program tar vytváření inkrementálních záloh. Prostřednictvím této volby zadáte datum a tar s tímto datem porovnává čas modifikace všech zadaných souborů. Je-li čas modifikace novější než zadané datum, bude soubor zahrnut do zálohy. Následující příkaz bere datum z časové značky předchozího archivu. Nejprve vytvoříme iniciální (první) archiv a zobrazíme jeho časovou značku. Následně vytvoříme nový soubor a poté vytvoříme novou zálohu, do níž bude zahrnut jen tento nový soubor:

```
jimmy:~> tar cvpf /var/tmp/javaproggies.tar java/*.java java/btw.java java/error.java java/hello.java java/income2.java java/income.java
java/inputdevice.java java/input.java java/master.java java/method1.java java/mood.java java/moodywaitress.java java/test3.java
java/TestOne.java java/TestTwo.java java/Vehicle.java

jimmy:~> ls -l /var/tmp/javaproggies.tar -rw-rw-r-- 1 jimmy jimmy 10240 Jan 21 11:58 /var/tmp/javaproggies.tar

jimmy:~> touch java/newprog.java

jimmy:~> tar -N /var/tmp/javaproggies.tar \ -cvp /var/tmp/incremental1-javaproggies.tar java/*.java 2> /dev/null
java/newprog.java

jimmy:~> cd /var/tmp/

jimmy:~> tar xvf incremental1-javaproggies.tar java/newprog.java
```

Standardní chyby jsou směrovány do /dev/null. Pokud byste to neudělali, tar pro každý nezmě-něný soubor vypíše zprávu oznamující, že tento soubor nebude do zálohy zahrnut.
Tento způsob práce má nevýhodu v tom, že kontroluje časové značky souborů. Řekněme, že si do zálohovaného adresáře stáhnete a rozbalíte nějaký archiv, který obsahuje soubory staré třeba dva roky. Při příštím zálohování budou časové značky těchto souborů porovnány s iniciálním archivem a nové soubory budou považovány za příliš staré, a tar je tedy do inkrementální zálo-hy nezahrne.

Lepší variantou je použít volbu -g, která vytvoří seznam zálohovaných souborů. Při inkrementál-ním zálohování se pak soubory porovnávají s tímto seznamem. Funguje to takto:

```
jimmy:~> tar cvpf work-20030121.tar -g snapshot-20030121 work/ work/ work/file1 work/file2 work/file3

jimmy:~> file snapshot-20030121 snapshot-20030121: ASCII text
```

Další den *jimmy* změní file3 a vytvoří nový file4. Na konci dne pořizuje novou zálohu:

```
jimmy:~> tar cvpf work-20030122.tar -g snapshot-20030121 work/work/work/file3work/file4
```

Tyto příklady byly velmi jednoduché, jejich princip však můžete použít pro vytvoření úloh naplá-novaným prostřednictvím *cronu* (viz kapitolu „Cron a crontab“). Automaticky tak můžete vytvářet úplné týdenní zálohy a inkrementální denní zálohy. Při vytváření plné zálohy je nutné přepsat seznam zálohovaných souborů, používaný následně k inkrementálním zálohám.

Další informace naleznete v dokumentaci k příkazu tar.

> Opravdový život Jak jste už asi postřehli, je tar výhodný nástroj při archivaci adresářů, tedy souborů, které logicky patří k sobě. Pokud byste chtěli archivovat celé diskové oddíly, disky či velké pro-jekty, existují vhodnější nástroje. Hovořili jsme o programu tar zejména proto, že jde
> o velmi oblíbený nástroj pro distribuci archivů. Velmi často se stává, že instalujete program, který je distribuován jako „komprimovaný tarball“. Jednodušší způsoby pravidelného zálo-hování systému popisujeme v kapitole „Použití programu rsync“.

Komprimace a dekomprimace programy gzip nebo bzip2

Data, včetně tarballů, je možné komprimovat pomocí komprimačních nástrojů. Příkaz gzip přidá k názvu souboru příponu *.gz* a odstraní původní soubor.

```
jimmy:~> ls -la | grep tar -rw-rw-r-- 1 jimmy jimmy 61440 Jun 6 14:08 obrazky-bez-adresare.tar

jimmy:~> gzip obrazky-bez-adresare.tar

jimmy:~> ls -la obrazky-bez-adresare.tar.gz -rw-rw-r-- 1 jimmy jimmy 50562 Jun 6 14:08 obrazky-bez-adresare.tar.gz
```

Dekomprimaci souboru provedete volbou -d.
Program bzip2 funguje podobně, používá však vylepšený komprimační algoritmus a výsledkem jsou proto menší soubory. Více informací naleznete na informačních stránkách programu bzip2. Softwarové balíky pro Linux se typicky distribuují jako „gzipované tarbally“. Po rozbalení takové-ho archivu je vždy rozumné najít soubor README a přečíst si jej. Obvykle obsahuje návod pro instalaci balíku.
GNU verze programu tar umí přímo pracovat s gzipovanými soubory. Příkazem

```
tar zxvf file.tar.gz
```

provedete dekomprimaci a rozbalení archivu .tar.gz nebo .tgz. Příkazem

```
tar jxvf file.tar.bz2
```

rozbalíte archiv komprimovaný programem bzip2.

Javové archivy

Projekt GNU nabízí nástroj jar pro vytváření javových archivů. Jde o javovou aplikaci, která kom-binuje více souborů do jediného archivního souboru JAR. I když jde o obecný archivační a kom-primační nástroj, založený na komprimačních formátech ZIP a ZLIB, je jar primárně určen k zaba-lení javového kódu, appletů a/nebo aplikací do jediného souboru. Komponenty javové aplikace zabalené v jednom archivu se stáhnou podstatně rychleji.

Na rozdíl od programu tar komprimuje program jar automaticky, nezávisle na ostatních nástro-jích – jde totiž v zásadě o javovou verzi programu zip. Navíc umožňuje jednotlivé soubory v archi-vu podepsat, takže je možné ověřit jejich původ.

Syntaxe je prakticky shodná s programem tar, konkrétní rozdíly naleznete na informačních strán-kách programu.

> Upozornění: tar, jar a symbolické odkazy
>
> Ve standardní dokumentaci není zmíněna jedna zaznamenání hodná funkce, a to ta, že jar následuje symbolické odkazy. Data, na něž odkazy míří, budou zahrnuta do archivu. Pro gram tar se standardně chová tak, že zálohuje pouze symbolický odkaz, nikoliv jeho cíl. Toto chování je ovšem možné změnit přepínačem -h.

Přenos dat

Uložení kopie dat na jiném počítači je jednoduchá a vyhovující metoda zálohování. Informace

o příkazech jako scp, ftp a dalších naleznete v kapitole „Sítě“. V další části se budeme věnovat

lokálním zálohovacím zařízením.

Přesun dat na zálohovací zařízení

Kopírování na disketu Formátování diskety

Na většině linuxových systémů má uživatel možnost použít disketovou jednotku. Název zařízení závisí na velikosti a počtu disketových mechanik, v případě nejasností se obraťte na správce systé-mu. Na rozumně spravovaných systémech bude nejspíš existovat odkaz /dev/floppy, mířící na /dev/fd0 (zařízení s autodetekcí typu média) nebo na /dev/fd0H1440 (disketová mechanika s kapacitou 1,44 MB).
K nízkoúrovňovému formátování disket slouží příkaz fdformat. Jako parametr se mu zadává název disketového zařízení. Je-li disketa chráněna proti zápisu, vypíše fdformat chybové hlášení.

emma:~> fdformat /dev/fd0H1440 Double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB. Formatting ... done Verifying ... done emma:~>

K vytvoření diskety ve formátu kompatibilním se systémem DOS slouží příkaz mformat (z balíku mtools). S takovou disketou je možné následně pracovat pomocí příkazů mcopy, mdir a dalších. K dispozici jsou i grafické nástroje.

Po naformátování je disketu možné připojit do souborového systému a pracovat s ní jako s nor
málním (byť malým) adresářem – obvykle se diskety připojují do adresáře /mnt/floppy. Téměř všechny linuxové distribuce obsahují nástroj mkbootdisk, kterým vytvoříte bootovací systémovou disketu.

Přenos dat příkazem dd

Příkazem dd můžete data zapsat na disk nebo je z něj přečíst, podle toho, jaká zadáte vstupní a výstupní zařízení. Například:

gaby:~> dd if=obrazky-bez-adresare.tar.gz of=/dev/fd0H1440 98+1 records in 98+1 records out

gaby~> dd if=/dev/fd0H1440 of=/var/tmp/obrazky.tar.gz 2880+0 records in 2880+0 records out

gaby:~> ls /var/tmp/obrazky* /var/tmp/obrazky.tar.gz
Přenos dat se provádí nad nepřipojeným zařízením. Takto vytvoření diskety nebude možno při-pojit do souborového systému, přesně tímto způsobem se ale vytvářejí bootovací nebo záchran-né diskety. Další možnosti příkazu dd naleznete na jeho manuálové stránce.

Program je součástí GNU balíku *coreutils*.

> Obraz disku Příkazem dd můžete vytvořit také nízkoúrovňový obraz celého disku. Například na note-booku mám na oddílu /dev/hda1 nainstalován proprietární operační systém. Po jeho ini-ciální instalaci, aktualizaci a nastavení jsem celý oddíl zálohoval příkazem dd if=/dev/hda1 of=win.img. Místo rituální reinstalace uvádím tento systém do použitelné-ho stavu příkazem dd if=win.img of=/dev/hda1. Rychlé a pohodlné.

Vytvoření kopie na CD vypalovačce

Na některých systémech mohou uživatelé používat i CD vypalovačku. Nejprve je nutné připravit data. V adresáři, jehož obsah chcete zálohovat, spusťte příkaz mkisofs. Příkazem df si ověřte, že je na disku dostatek místa, protože vznikne soubor s velikostí odpovídající celému zálohovanému adresáři:

[rose@blob recordables] df -h . Filesystem Size Used Avail Use% Mounted on /dev/hde5 19G 15G 3.2G 82% /home

[rose@blob recordables] du -h -s . 325M .

[rose@blob recordables] mkisofs -J -r -o cd.iso . <--snap--> making a lot of conversions <--/snap--> 98.95% done, estimate finish Fri Apr 5 13:54:25 2002 Total translation table size: 0 Total rockridge attributes bytes: 35971 Total directory bytes: 94208 Path table size(bytes): 452 Max brk space used 37e84 166768 extents written (325 Mb)

Volby -J a -r slouží k vytvoření disku, který bude možno připojit i na jiných operačních systémech, podrobnosti naleznete na manuálových stránkách. Následně můžete příkazem cdrecord s vhod-nými parametry vypálit CD.

[rose@blob recordables] cdrecord -dev 0,0,0 -speed=8 cd.iso Cdrecord 1.10 (i686-pc-linux-gnu) (C) 1995-2001 Joerg Schilling scsidev: '0,0,0' scsibus: 0 target: 0 lun: 0 Linux sg driver version: 3.1.20 Using libscg version 'schily-0.5' Device type : Removable CD-ROM Version : 0 Response Format: 1 Vendor_info : 'HP '
Identification : 'CD-Writer+ 8100 'Revision : '1.0g'Device seems to be: Generic mmc CD-RW.Using generic SCSI-3/mmc CD-R driver (mmc_cdr).Driver flags : SWABAUDIOStarting to write CD/DVD at speed 4 in write mode for single session.Last chance to quit, starting real write in 0 seconds. Operation starts.

V závislosti na parametrech vypalovačky máte nyní čas na cigaretu a/nebo kávu. Po skončení ope-race se vypíše potvrzující

hlášení:

Track 01: Total bytes read/written: 341540864/341540864 (166768 sectors).

U novějších verzí programu cdrecord se parametr dev zadává trochu jinak: cdrecord dev=ATA:0,0,0… Detaily najdete v manuálových stránkách programu cdrecord. V době DVD vypalovaček vás bude zajímat také příkaz growisofs, který vytvoří obraz DVD a hned jej vypálí.

K usnadnění celé operace existuje mnoho grafických nástrojů. Oblíbeným je například xcdroast, který je volně k dispozici na adrese *http://www.xcdroast.org/* a je součástí většiny distribucí i GNU adresáře. Prostředí KDE a Gnome rovněž obsahují vlastní nástroje pro vypalování CD, například jednoduchý a intuitivní program K3B.

Zálohování na/z mechaniky Jazz, USB zařízení a podobně

Tato zařízení bývají typicky standardním způsobem připojena do souborového systému. Po jejich připojení s nimi pracujete jako s běžnými adresáři, takže můžete použít standardní příkazy pro manipulaci se soubory.

Následující příklad ukazuje zkopírování obrázků z USB fotoaparátu na disk:

```
robin:~> mount /mnt/camera

robin:~> mount | grep camera/dev/sda1 on /mnt/camera type vfat (rw,nosuid,nodev)
```

Takovéto použití je bezpečné pouze v případě, že je fotoaparát jediným úložným USB zařízením, které k systému připojujete. Nezapomínejte, že USB zařízením se položky v adresáři /dev přidě-lují v tom pořadí, v jakém zařízení připojujete k systému. Pokud tedy jako první připojíte USB klí-čenku, bude připojena jako zařízení /dev/sda, pokud následně připojíte fotoaparát, připojí se jako /dev/sdb – a to samozřejmě za předpokladu, že nemáte nainstalovány žádné SCSI disky, které se také připojují jako /dev/sd*. Na novějších systémech s jádrem 2.6 se používá hotplug systém HAL (Hardware Abstraction Layer), který uživatele od těchto podrobností izoluje. Potřebujete-li zjistit, jako co se dané zařízení připojilo, zadejte příkaz dmesg.

Nyní tedy můžeme kopírovat soubory:

```
robin:~> cp -R /mnt/camera/* images/

robin:~> umount /mnt/camera
```

Mechaniky Jazz se analogicky připojují jako /mnt/jazz.Aby vše správně fungovalo, musíte mít v souborech /etc/modules.conf a /etc/fstab používaná zaří-zení správně nastavena. Podrobnosti naleznete v HOWTO dokumentech ke specifickému hard-warovému zařízení.

Zálohování dat na pásková zařízení

Provádí se příkazem tar (viz výše). K obsluze magnetických pásek (jako /dev/st0) slouží nástroj mt. O páskovém zálohování byly napsány celé knihy, podrobnější informace naleznete v publi-kacích uvedených v příloze B. Při zálohování databází může být v důsledku jejich architektury nutné použít zcela jiné zálohovací postupy.

Příkazy pro automatické zálohování se typicky instalují jako úlohy *cronu*, aby tak bylo zajištěno jejich automatické pravidelné spouštění. Ve větších prostředích je možné zálohovat více počítačů buď pomocí volně dostupného zálohovacího nástroje Amanda nebo pomocí komerčních záloho-vacích systémů. Zálohování na pásky je oblast mimo záběr tohoto dokumentu.

Nástroje vaší distribuce

Většina distribucí nabízí vlastní zálohovací nástroje. Například: ■ openSUSE: YaST obsahuje moduly

pro zálohování a obnovu dat.

RedHat: Správce souborů umožňuje grafickou manipulaci s (komprimovanými) archivy. K přesunu záloh na externí zařízení je zajištěna integrace s nástroje X-CD-Roast.
Mandriva Linux: Obsahuje vlastní nástroj pro zálohování a obnovu dat DrakBackup.
Většina distribucí obsahuje BSD nástroje dump a restore, které umožňují zálohovat sou-borové systémy *ext2* a *ext3*. Tyto nástroje dokážou zapisovat na mnoho různých zařízení a v zásadě fungují tak, že na cílové zařízení zapíší zálohovaný soubor či souborový systém bit po bitu. Stejně jako u programu dd je tak možno zálohovat i speciální soubory, jako jsou například zařízení v /dev.

Použití programu rsync

Úvod

Program rsync je rychlý a pružný nástroj pro vzdálené zálohování. Na unixových systémech je velmi rozšířený, snadno se nastavuje a používá ve skriptech. Písmeno *r* v názvu rsync pochází ze slova *remote*, nemusí to však platit doslova. Jako

„vzdálené“ zařízení můžete použít i USB úložiš-tě nebo jiný diskový oddíl, nemusí jít o jiný počítač.

Příklad: rsync na USB úložiště

Jak už jsme říkali v kapitole „Přípojné body“, je nutné zařízení nejprve připojit. To musíme pro-vést jako *root*:

```
root@theserver# mkdir /mnt/usbstore

root@theserver# mount -t vfat /dev/sda1 /mnt/usbstore
```

Je samozřejmě nutné, aby váš systém USB zařízení podporoval. Pokud vám podpora USB zaříze-ní nefunguje, použijte USB příručku na adrese *http://www.linux-usb.org/USB-guide/*. Příkazem dmesg ověřte, zda se USB zařízení skutečně hlásí jako dev/sda1.

Nyní můžete zahájit zálohování, například adresáře /home/karl:

```
karl@theserver:~> rsync -avg /home/karl /mnt/usbstore
```

Podrobnosti naleznete jako vždy na příslušných manuálových stránkách.

Shrnutí

Následující tabulka uvádí seznam příkazů, které se vztahují k zálohování:

Příkaz	Popis
	Komprimační a dekomprimační nástroj.
	Vypaluje zvuková nebo datová CD.
	Konvertuje a kopíruje soubory.
	Nízkoúrovňové formátování disket.
	Komprimační a dekomprimační nástroj.
	Kopíruje soubory ve formátu MS-DOS na/z unixového systému.
	Vypíše adresář ve formátu MS-DOS.
	Na nízkoúrovňově naformátované disketě vytvoří souborový systém MS-DOS.
	Vytvoří bootovací disketu aktuálního systému.
	Připojuje souborové systémy. (Integruje je do stávajícího souborového systému v místě připojení.)
	Synchronizuje obsah adresářů.
	Páskový archivační nástroj, používá se také k vytváření archivů na disku.
	Odpojuje souborové systémy.

Zálohovací příkazy

Cvičení

Příkazem tar vytvořte v adresáři /var/tmp zálohu svého domovského adresáře. Tu násled-ně zkomprimujte příkazem gzip nebo bzip2. Vytvořte korektní archiv tak, aby při jeho roz-balení nevznikl nepořádek.
Naformátujte disketu a zkopírujte na ni nějaké soubory ze svého domovského adresáře. Disketu si vyměňte s kolegou a ve svém domovském adresáři obnovte soubory z jeho diskety.
Naformátujte disketu pro systém MS-DOS. Pomocí nástrojů z balíku *mtools* na ni zkopíruj-te a smažte soubory.
Co se stane, pokusíte-li se k souborovému systému připojit nenaformátovanou disketu?
Máte-li USB klíčenku, zkuste na ni zkopírovat nějaký soubor.
Pomocí příkazu rsync vytvořte kopii svého domovského adresáře na jiném lokálním nebo vzdáleném souborovém systému.

Sítě

Jakmile dojde na počítačové sítě, je Linux tou pravou volbou. Nejen že jsou síťové funkce úzce integrovány přímo do jádra operačního systému a je k dispozici celá řada volně dostupných nástrojů a aplikací, ale silnou stránkou Linuxu je také jeho robustnost pod vysokou zátěží, která je důsledkem mnoha let ladění a testování tohoto otevřeného projektu.

O Linuxu a sítích byly napsány celé řady knih, v této kapitole se pokusíme nastínit základní pře-hled. Po jejím prostudování budete vědět více o následujících tématech:

Podporované síťové protokoly
Konfigurační soubory sítě
Příkazy pro konfiguraci a testování sítě
Démony a klientské programy různých síťových aplikací
Sdílení a tisk souborů
Vzdálené spouštění příkazů a aplikací
Základy propojování sítí
Bezpečné spouštění vzdálených aplikací
Firewally a detekce útoků

Základní přehled

Síové protokoly

Protokol je, jednoduše řečeno, sada pravidel pro vzájemnou komunikaci.

Linux podporuje mnoho různých síťových protokolů. Zmíníme se jen o těch nejdůležitějších:

TCP/IP

Protokoly Transport Control Protocol a Internet Protocol jsou dva nejrozšířenější způsoby komu-nikace na Internetu. Na těchto protokolech je vystavěna celá řada aplikací, například webové pro-hlížeče a poštovní programy.

Zjednodušeně řečeno nabízí protokol IP metodu pro zasílání paketů s informacemi z jednoho počítače na druhý, zatímco TCP zajistí, že pakety jsou organizovány do datových proudů, takže nedojde k promíchání paketů různých aplikací a pakety budou odeslány a přijaty ve správném pořadí.

Internetové protokoly byly původně vyvinuty před třiceti lety pro americké ministerstvo obrany, zejména za účelem spolehlivého propojení počítačů různých výrobců. Dalším důvodem vývoje bylo vytvoření spolehlivého transportního systému na nespolehlivé síti.
Protokolová rodina TCP/IP je v Linuxu podporována už od jeho začátků. Byla implementována úplně nezávisle a jde o jednu z nejrobustnějších, nejrychlejších a nejspolehlivějších implementací této rodiny, což je jeden z klíčových faktorů úspěchu Linuxu. Linux a sítě spolu natolik souvisí, že například bootování počítače nepřipojeného k síti může trvat mnohem déle a může vést k řadě problémů. Dokonce i pokud počítač vůbec není k síti připojen, používají se síťové protokoly pro interní komunikaci systému a aplikací. Linux počítá s tím, že bude připojen k síti.

Dobrý začátek k získání informací o protokolech TCP a IP představují následující dokumenty:

man 7 ip: Popisuje implementaci IPv4 v Linuxu (verze 4 je momentálně nejrozšířenější verzí IP protokolu).
man 7 tcp: Implementace protokolu TCP.
RFC793, RFC1122, RFC2001 pro TCP a RFC791, RFC1122 a RFC1112 pro IP.

Dokumenty RFC obsahují popisy síťových standardů, protokolů, aplikací a implementací. Doku-menty udržuje IETF (Internet Engineering Task Force) a mezinárodní komunita s cílem umožnit hladký chod Internetu a vývoj internetové architektury.

Archiv RFC dokumentů je k dispozici na mnoha místech Internetu, například na adrese
http://www.ietf.org/rfc.html.

TCP/IPv6

Nikdo neočekával, že k rozvoji Internetu dojde takovým tempem, jakým k němu došlo. V přípa-dě propojení obrovského počtu počítačů do jedné sítě se ukázalo několik slabých míst protokolu IP – zejména nedostatek unikátních adres, přidělovaných jednotlivým připojeným zařízením. Proto byla uvedena verze 6 protokolu IP, která řeší potřeby dnešního (a budoucího) Internetu.

Naneštěstí protokol IPv6 doposud nepodporují všechny aplikace a služby. V mnoha prostředích, pro něž je upgrade na verzi 6 výhodný, nyní probíhá postupná migrace. Některé aplikace dopo-sud používají starší verzi protokolu, jiné aplikace už pracují s novou verzí. Při manipulaci s kon-figurací sítě to může být občas problém, protože do hry vstupují různá opatření, která obě dvě verze protokolu vzájemně skrývají tak, aby nedocházelo ke vzájemnému míchání spojení těchto dvou verzí.

Více informací můžete nalézt v Ipv6 HOWTO v této knize, případně v následujících dokumentech:

man 7 ipv6: Implementace IPv6 v Linuxu,

RFC1883 definující protokol Ipv6.

PPP, SLIP, PLIP, PPPoE

Linuxové jádro obsahuje vestavěnou podporu protokolů PPP (Point-to-Point-Protocol), SLIP (Seri-al Line IP) a PLIP (Parallel Line IP). Protokol PPP představuje nejčastější metodu, kterou se indi-viduální uživatelé připojují ke svému ISP (hovoříme o vytáčeném připojení). Postupně tento pro-tokol ustupuje protokolu PPPoE (PPP over Ethernet), který se používá při připojení kabelovým modemem.

Většina linuxových distribucí nabízí snadno použitelné nástroje pro konfiguraci internetového při-pojení. V případě vytáčeného připojení tak v zásadě potřebujete znát pouze jméno, heslo a tele-fonní číslo, které vám sdělil váš ISP. Tyto údaje zadáte do grafického konfiguračního nástroje, který vám následně umožní navázat a ukončit spojení s poskytovatelem.

ISDN

Jádro Linuxu obsahuje rovněž vestavěnou podporu ISDN. Projekt Isdn4linux umožňuje používat ISDN PC karty a dokáže emulovat modem se standardními Hayes příkazy („AT“ příkazy). Díky tomu je možné použít cokoliv od terminálového programu až po plnohodnotné připojení k Internetu.

Podrobnosti naleznete v dokumentaci k systému.

AppleTalk

AppleTalk je název protokolové rodiny společnosti Apple. Slouží k vytváření peer-to-peer sítí se základními službami, jako je sdílení souborů a tiskáren. Každý počítač může současně působit jako server i jako klient, potřebný hardware a software je součástí každého počítače Apple.

Linux obsahuje úplnou podporu sítí AppleTalk. Netatalk je jaderná implementace protokolové rodiny AppleTalk, založená na BSD implementaci. Obsahuje podporu směrování, umožňuje pro-střednictvím AppleShare sdílet unixové souborové systémy a nabízet unixové tiskárny a využívat AppleTalk tiskáren.

SMB/NMB

Kvůli kompatibilitě se sítěmi MS Windows je možné na unixových strojích nainstalovat balík Samba, který zahrnuje podporu protokolů NMB a SMB/CIFS. Protokol SMB (označovaný též jako Server Message Block, Session Message Block, NetBIOS či LanManager) se v systémech MS Win-dows 3.11, NT, 95/98, 2K a XP používá ke sdílení disků a tiskáren.

Mezi základní funkce balíku Samba patří: poskytnutí linuxových disků a tiskáren pro sdílení z Win
dows a sdílení disků a tiskáren nabízených z Windows v Linuxu. Většina linuxových distribucí obsahuje balík *samba*, který zajistí větší část nastavení serveru a při startu počítače standardně spustí démony smbd, server Samba a nmbd, jmenný server NetBIOSu. Sambu je možné nastavovat v grafickém prostředí, přes webové rozhraní anebo v příkazovém řádku prostřednictvím textových konfiguračních souborů. Zmíněné démony způsobí, že se počí-tač s Linuxem objevuje v oknech Síťové okolí počítačů s Windows a jím nabízené sdílené pro-středky se chovají stejně jako prostředky nabízené jinými systémy Windows.

Podrobnější informace naleznete na následujících místech:

man smb.conf: popisuje formát hlavního konfiguračního souboru systému Samba.
Dokumentace projektu Samba na adrese *http://us4.samba.org/samba/*.

Ostatní protokoly

Linux podporuje i řadu dalších protokolů, například Amateur Radio, různé WAN protokoly (X25, FrameRelay, ATM), infračervené a další bezdrátové protokoly. Jejich použití je typicky vázáno na specializovaný hardware, proto o nich nebudeme v tomto dokumentu hovořit.

Konfigurace sítě a informace o síti Konfigurace lokálních síových rozhraní

Všechny ty velké, uživatelsky příjemné linuxové distribuce obsahují různé grafické nástroje, které umožňují snadno nastavit parametry připojení k lokální síti nebo k poskytovateli internetového připojení. Tyto nástroje se spouští buď z příkazového řádku nebo z menu.

- RedHat Linux obsahuje nástroj redhat-config-network, který má grafické i textové roz-hraní.
- openSUSE obsahuje univerzální konfigurační nástroj YAST nebo YAST2.
- Mandrake/Mandriva Linux obsahuje „Network and Internet Configuration Wizard“, který se typicky spouští z Ovládacího centra Mandriva.

Způsob použití a další informace o příslušných nástrojích naleznete v dokumentaci ke své distri-buci.K nastavení sítě potřebujete znát následující informace:

Při připojení k lokální síti, například doma nebo v práci: název počítače, název domény a IP adresu. Chcete-li sestavit vlastní síť, doporučujeme vám nejprve si o problematice něco přečíst. V práci je pravděpodobné, že počítač bude rovnou správně nastaven. Pokud si nejste jisti, raději nezadávejte nic, než abyste si vymýšleli vlastní hodnoty.
Při připojení k Internetu: jméno a heslo pro připojení k ISP, telefonní číslo, používáte-li modem. Název počítače, IP adresu a další údaje potřebné pro správnou funkci internetových aplikací vám obvykle přímo nastaví váš ISP.

Konfigurační soubory sítě

Grafické konfigurační nástroje prostřednictvím několika základních příkazů modifikují několik základních konfiguračních souborů sítě. Přesné názvy těchto souborů a jejich umístění v soubo-rovém systému hodně závisí na konkrétní distribuci a verzi. Nicméně některé konfigurační sou-bory jsou společné ve všech unixových systémech:

- /etc/hosts: Vždy definuje adresu 127.0.0.1 a její název *localhost*, která se používá při mezi-procesové komunikaci. Tento řádek nikdy neodstraňujte! Někdy obsahuje i adresy dalších systémů, které jsou pak dostupné i bez využití externí jmenné služby, jako je například DNS (Domain Name System).

  Příklad souboru hosts v malé síti by mohl vypadat takto:

  ```
  # Do not remove the following line, or various programs
  # that require network functionality will fail.

  127.0.0.1 localhost.localdomain localhost
  192.168.52.10 tux.mylan.com tux
  192.168.52.11 winxp.mylan.com winxp
  ```

  Více informací získáte příkazem man hosts.

- /etc/resolv.conf: Nastavuje přístup k serverům DNS. Tento soubor definuje implicitně pro-hledávané domény a adresy serverů DNS:

  ```
  search mylan.com
  nameserver 193.134.20.4
  ```

  Více informací získáte příkazem man resolv.conf.

- /etc/nsswitch.conf: Definuje pořadí, ve kterém se mají používat různé jmenné služby. Jste--li připojení k Internetu, je nutné, aby byla v souboru na řádku „hosts“ uvedena položka *dns*:

  ```
  [bob@tux ~] grep hosts /etc/nsswitch.confhosts: files dns
  ```

  Toto nastavení počítači říká, aby názvy a IP adresy nejprve hledal v souboru /etc/hosts, a teprve pokud zde neuspěje, aby se obrátil na servery DNS. Další možné jmenné služby jsou LDAP, NIS a NIS+.

  Více informací viz man nsswitch.conf.

Příkazy pro konfiguraci sítě

Různé skripty a grafické nástroje používané v různých distribucích nastavují jaderné konfigurační údaje sítě pomocí příkazu ip, případně (na starších systémech) pomocí příkazů ifconfig a route. Příkaz ip se používá k nastavení IP adresy rozhraní, k nastavování směrovacích údajů, ke zobra-zení nastavených parametrů a k celé řadě dalších úkonů. Následující příkazy vypíší nastavení IP adres a směrovacích údajů:

```
benny@home benny> ip addr show
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00 inet
    127.0.0.1/8 brd 127.255.255.255 scope host lo inet6 ::1/128 scope host
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100 link/ether 00:50:bf:7e:54:9a brd ff:ff:ff:ff:ff:ff
    inet 192.168.42.15/24 brd 192.168.42.255 scope global eth0 inet6 fe80::250:bfff:fe7e:549a/10 scope link

benny@home benny> ip route show 192.168.42.0/24 dev eth0 scope link
127.0.0.0/8 dev lo scope link default via 192.168.42.1 dev eth0
```

Máte-li notebook, který většinou připojujete k firemní síti prostřednictvím vestavěného etherneto-vého adaptéru, avšak nyní na něm potřebujete nastavit vytáčené připojení pro přístup z domu či z hotelu, budete možná muset rozběhnout PCMCIA kartu. K tomu slouží nástroj cardctl. Kvalitní distribuce většinou umožňují nastavit PCMCIA karty přímo z konfigurátoru sítě, takže uživatel nebude nucen spouštět příkazy pro její nastavení ručně.

Podrobnější popis konfigurace sítě je mimo záběr tohoto dokumentu. Primárním zdrojem dalších informací jsou vždy manuálové stránky služeb, které chcete nastavit. Další informace naleznete mimo jiné v těchto dokumentech:

Modem HOWTO, *http://www.tldp.org/HOWTO/Modem-HOWTO.html.* Popisuje výběr, při-pojení, nastavení a diagnostiku

analogových modemů.
HOWTO dokumenty z kategorie sítí, *http://www.tldp.org/HOWTO/HOWTO-INDEX/ networking.html.* Zde naleznete celou řadu dokumentů o sítích obecně, o různých proto-kolech, vytáčeném připojení, DNS, VPN, přepínání, směrování, bezpečnosti a dalších.
Soubor /usr/share/doc/iproute-<version>ip-cref.ps, který můžete zobrazit například prohlí-žečem gv.

Názvy síových rozhraní

Rozhraní *lo* nebo *local loop* je na linuxovém systému vždy spojeno s interní adresou 127.0.0.1. Pokud by toto zařízení neexistovalo, nefungovala by správně celá řada aplikací. Proto je v systé-mu vždy najdete, i pokud počítač není připojen k síti.

První ethernetové zařízení, *eth0* v případě standardní síťové karty, bývá připojeno k lokální síti. Typický uživatelský počítač má pouze jedinou síťovou kartu. Směrovače, které propojují sítě mezi sebou, mají jedno síťové rozhraní pro každou obsluhovanou síť.

Pokud se k Internetu připojujete modemem, síťové zařízení se bude pravděpodobně jmenovat *ppp0*. Obvykle to platí i v případě, že se připojujete kabelovým modemem.

Nastavení vašeho systému

Kromě zobrazení síťových nastavení příkazem ip můžete celou řadu informací zjistit také příka-zem netstat, který nabízí celou řadu voleb, a obecně jde o velmi užitečný příkaz. Volba -i slouží k výpisu informací o síťových rozhraních:

bob:~> netstat -i Kernel Interface table Iface MTU Met RXOK RXERR RX-DRP RX-OVR TX-OK TX-ERR TX-DRP TX-OVR Flg

eth0 1500 0 58459 0 0 0 0 0 63865 0 0 0 BMRU
lo 16436 24060 0 0 24060 0 0 0 LRU

Směrovací údaje můžete vypsat volbou -nr:

bob:~> netstat -nr Kernel IP routing table Destination Gateway Genmask Flags MSS Window irtt Iface

192.168.42.0 0.0.0.0 255.255.255.0 U 40 0 0 eth0
127.0.0.0 0.0.0.0 255.0.0.0 U 40 0 0 lo
0.0.0.0 192.168.42.1 0.0.0.0 UG 40 0 0 eth0

Jde o typický uživatelský počítač připojení k IP síti. Má pouze jediné síťové rozhraní, *eth0*. Roz-hraní *lo* je lokální.

Jiné počítače

Pro správu sítě a vzdálenou administraci linuxových strojů je k dispozici úctyhodné množství nástrojů. Celou řadu jich najdete na každém serveru s linuxovou tematikou. V tomto dokumentu nemáme prostor na podrobný popis různých nástrojů, vždy proto pracujte s dokumentací konkrétního programu.
V dalším textu se zmíníme jen o několika základních unixových/linuxových textových nástrojích. Informace o doménách, názvech a adresách počítačů zjistíte příkazem host:

[emmy@pc10 emmy]$ host www.eunet.bewww.eunet.be. has address 193.74.208.177

[emmy@pc10 emmy]$ host -t any eunet.beeunet.be. SOA dns.eunet.be. hostmaster.Belgium.EU.net. 2002021300 28800 7200 604800 86400 eunet.be. mail is handled by 50 pophost.eunet.be. eunet.be. name server ns.EU.net. eunet.be. name server dns.eunet.be.

Podobné údaje můžete zjistit také příkazem dig, který vám navíc poskytne více informací o tom, jak jsou záznamy na jmenných serverech uloženy.Zda je konkrétní počítač dostupný, můžete ověřit příkazem ping. Pokud váš systém odesílá vícepaketů, můžete ping přerušit stiskem kláves Ctrl+C:

[emmy@pc10 emmy]$ ping a.host.be PING a.host.be (1.2.8.3) from 80.20.84.26: 56(84) bytes of data. 64 bytes from a.host.be(1.2.8.3):icmp_seq=0 ttl=244 time=99.977msec --- a.host.be ping statistics ---1 packets transmitted, 1 packets received, 0% packet loss round-trip min/avg/max/mdev = 99.977/99.977/99.977/0.000 ms

Chcete-li ověřit, kudy pakety k nějakému systému putují, použijte příkaz traceroute:
[emmy@pc10 emmy]$ /usr/sbin/traceroute www.eunet.be traceroute to www.eunet.be(193.74.208.177),30 hops max,38b packets 1 blob (10.0.0.1)
0.297ms 0.257ms 0.174ms 2 adsl-65.myprovider.be (217.136.111.1) 12.120ms 13.058ms 13.009ms 3 194.78.255.177 (194.78.255.177) 13.845ms 14.308ms 12.756ms 4

gigabitethernet2-2.intl2.gam.brussels.skynet.be (195.238.2.226)
13.123ms 13.164ms 12.527ms
5 pecbru2.car.belbone.be (194.78.255.118)
16.336ms 13.889ms 13.028ms

6 ser-2-1-110-ias-be-vil-ar01.kpnbelgium.be (194.119.224.9) 14.602ms 15.546ms
15.959ms 7 unknown-195-207-939.eunet.be (195.207.93.49) 16.514ms
17.661ms 18.889ms 8 S0-1-0.Leuven.Belgium.EU.net (195.207.129.1)
22.714ms 19.193ms 18.432ms 9 dukat.Belgium.EU.net (193.74.208.178)
22.758ms * 25.263ms

Informace o konkrétních doménách můžete zjistit příkazem whois, který se dotazuje různých whois serverů:

[emmy@pc10 emmy]$ whois cnn.com [whois.crsnic.net]

Whois Server Version 1.3

<--zkráceno-->

Domain Name: CNN.COM
Registrar: NETWORK SOLUTIONS, INC.
Whois Server: whois.networksolutions.com
Referral URL: http://www.networksolutions.com
Name Server: TWDNS-01.NS.AOL.COM
Name Server: TWDNS-02.NS.AOL.COM
Name Server: TWDNS-03.NS.AOL.COM
Name Server: TWDNS-04.NS.AOL.COM
Updated Date: 12-mar-2002

>>> Last update of whois database: Fri, 5 Apr 2002 05:04:55 EST <<<

<--zkráceno-->

Record last updated on 12-Mar-2002.
Record expires on 23-Sep-2009.
Record created on 22-Sep-1993.
Database last updated on 4-Apr-2002 20:10:00 EST.

Domain servers in listed order:

TWDNS-01.NS.AOL.COM 149.174.213.151TWDNS-02.NS.AOL.COM 152.163.239.216TWDNS-03.NS.AOL.COM
205.188.146.88TWDNS-04.NS.AOL.COM 64.12.147.120

U jiných domén, než jsou .com, .net, .org a .edu, budete možná muset zadat, kterého serveru se dotazovat – pro české domény například takto:

whois *doména.cz@whois.nic.cz*

Internetové/intranetové aplikace

Linuxový systém představuje skvělou platformu pro poskytování síťových služeb. V této části uve-deme základní přehled nejběžnějších síťových serverů a aplikací.

Typy serverů Samostatné servery

Poskytování služeb uživatelům lze zajistit dvěma způsoby. Démon či služba může buď běžet jako samostatná aplikace anebo může být spouštěna prostřednictvím jiné služby. Síťové služby, které jsou hodně zatížené a/nebo trvale používané, typicky běží v samostatném reži-mu – jde o nezávislé trvale spuštěné programy. Nejčastěji se spouští hned při startu systému a pak čekají na příchozí požadavky na konkrétním připojovacím místě či portu tak, jak jsou nastaveny. Jakmile dorazí požadavek, je obsloužen a služba dál čeká na následující požadavky. Typickým pří-kladem je webový server: U něj chcete, aby byl k dispozici 24 hodin denně, a pokud je navíc hodně zatížen, může vytvářet více svých instancí, aby mohl současně obsluhovat více požadavků. Podobně fungují například také velké softwarové archivy jako Sourceforge (*http://sourceforge.net/*) nebo

Tucows (*http://tucows.com/*), které denně obsluhují tisíce požadavků.

Příkladem samostatné síťové služby na domácím počítači může být named, cachovací server DNS. Samostatně běžící služba má spuštěny své vlastní procesy, jak si můžete kdykoliv ověřit příkazem ps:

bob:~> ps auxw | grep named named 908 0.0 1.0 14876 5108 ? S Mar14 0:07 named -u named

U většiny služeb na domácím počítači, jako je například služba FTP, žádného spuštěného démo-na nenajdete, i tak ale můžete službu použít:

bob:~> ps auxw | grep ftpbob 738 690 0 16:17 pts/6 00:00:00 grep ftp

bob:~> ncftp localhostNcFTP 3.1.3 (Mar 27, 2002) by Mike Gleason (ncftp@ncftp.com).Connecting to localhost(127.0.0.1)...myhost.my.org FTP server (Version wu-2.6.2-8) ready.Logging in...Guest login ok, access restrictions apply.Logged in to localhost.ncftp / >

Způsob, jakým je to vyřešeno, si popíšeme v další části.

(x)inetd

Na domácím počítači obvykle nejde o tak mnoho. Můžete mít například malou síť, v níž čas od času z jednoho počítače na druhý kopírujete nějaké soubory prostřednictvím FTP nebo Samby (pokud potřebujete komunikovat i s MS Windows). V takovém případě by bylo čistě plýtváním prostředků, kdyby občas používané služby běžely trvale. V menších systémech proto obvykle všechny potřebné démony závisí na centrálním programu, který poslouchá na portech všech služeb, které obsluhuje.

Tento superserver či démon internetových služeb je spuštěn při startu systému. Existuje ve dvou běžných implementacích: inetd a xinetd (extended Internet services daemon). Jeden z nich obvykle najdete na každém linuxovém systému:

bob:~> ps -ef | grep inet root 926 1 0 Mar14 ? 00:00:00 xinetd-ipv6 -stayalive -reuse \ -pidfile /var/run/xinetd.pid

Služby, které se prostřednictvím tohoto démona spouštějí, jsou v případě inetd definovány v kon-figuračním souboru /etc/inetd.conf, v případě xinetd pak v adresáři /etc/xinetd.d. Mezi obvyklé služby, spouštěné tímto mechanismem, patří sdílení souborů a tiskáren, SSH, FTP, telnet, konfi-gurační démon Samby a časové služby.

Jakmile dojde k přijetí požadavku, centrální server spustí instanci serveru potřebné služby. Jakmi-le tedy v následujícím příkladu *bob* spustí FTP spojení s lokálním systémem, poběží FTP démon po dobu, kdy toto spojení trvá:

bob:~> ps auxw | grep ftp bob 793 0.1 0.2 3960 1076 pts/6 S 16:44 0:00 ncftp localhost ftp 794 0.7 0.5 5588 2608 ? SN 16:44 0:00 ftpd: localhost.localdomain: anonymous/bob@his.server.com: IDLE

Stejně to funguje, i pokud se připojujete na vzdálené systémy – buď vám démon odpoví přímo anebo vzdálený (x)inetd spustí vámi požadovanou službu a ukončí ji, jakmile s ní přestanete komunikovat.

Pošta Servery

Standardním poštovním serverem či MTA (Mail Transport Agentem) na unixových systémech je program *sendmail*. Jde o robustní škálovatelný program, který při správném nastavení na vhod-ném hardwaru bez potíží obslouží tisíce uživatelů. Podrobnější informace o nastavení Sendmailu jsou součástí balíků sendmail a sendmail-cf, zajímat vás mohou také soubory README a README.cf v adresáři /usr/share/doc/sendmail. Další užitečné informace se dozvíte na stránkách man sendmail a man aliases.

Dalším serverem je Qmail, který si získává popularitu tím, že se prohlašuje za bezpečnější než Sendmail. Zatímco Sendmail je jeden monolitický program, Qmail je tvořen několika menšími vzá-jemně komunikujícími programy, které je tak možné lépe zabezpečit. Poslední dobou získává obli-bu i poštovní server Postfix.

Oba tyto servery dokážou zajistit poštovní konference, filtraci pošty, virovou kontrolu a mnoho dalších. Poštovní konference je možné provozovat také prostřednictvím programů jako Mailman, Listserv, Majordomo nebo EZmlm. Pro účely virové kontroly je možné v Linuxu použít celou řadu jak komerčních, tak volně dostupných antivirových systémů. Podívejte se na webové stránky svého oblíbeného antivirového programu, zda nabízí i podporu linuxových serverů.

Vzdálené poštovní servery

Pro vzdálený přístup k poště se nejčastěji používají protokoly *POP3* a *IMAP*. Klienti těchto proto-kolů typicky umožňují pracovat offline a vzdáleně přistupovat k nové poště. Pro odesílání pošty pak používají protokol SMTP.
Protokol POP je poměrně jednoduchý, snadný na implementaci a podporuje jej většina poštov-ních klientů. Z následujících důvodů však doporučujeme používat protokol IMAP:

Dokáže udržovat trvalé příznaky stavu zpráv.
Dokáže zprávy ze serveru stahovat i je na něj ukládat.
Dokáže přistupovat k více poštovním schránkám.
Podporuje současné aktualizace a sdílené poštovní schránky.

Dokáže pracovat i s usenetovými zprávami a dalšími dokumenty.
Pracuje v on-line i off-line režimu.
Je optimalizován na on-line výkon zejména přes pomalé linky.

Poštovní klienti

Existuje celá řada textových i grafických poštovních klientů, zvaných též MUA (Mail User Agents).

Svého oblíbence si musíte zvolit sami.Už mnoho let je k dispozici unixový příkaz mail, který existoval ještě před vznikem počítačovýchsítí. Jde o jednoduché rozhraní k posílání zpráv a malých souborů jiným uživatelům, kteří mohouzprávu uložit, přesměrovat, odpovědět na ni a podobně.

I když už se tento program jako poštovní klient téměř nepoužívá, pořád může být užitečný, napří
klad pokud budete chtít někomu poslat výstup nějakého příkazu:

```
mail <future.employer@whereIwant2work.com> < cv.txt
```

Velkým vylepšením programu mail jsou poštovní klienti elm a pine (Pine Is Not ELM). Ještě novějším klientem je mutt, který obsahuje i takové funkce jako řazení zpráv do vláken.Pokud dáváte přednost grafickému rozhraní, máte na výběr z celých stovek možností. Nejoblíbe-nější volbou nových uživatelů je Mozilla Thunderbird nebo Evolution, klon MS Exchange, zná-zorněný na obrázku na následující straně.

Existují také desítky webmailových aplikací.

Přehled dostupných aplikací naleznete v dokumentu Linux Mail User HOWTO,

http://www.tldp.org/HOWTO/Mail-User-HOWTO/index.html.

Většina linuxových distribucí obsahuje program fetchmail, určený pro stahování a předávání pošty. Dokáže stahovat poštu ze vzdálených serverů (protokoly POP, IMAP a dalšími) a předat ji lokálnímu poštovnímu systému. S takto získanou poštou pak můžete pracovat pomocí normálních poštovních klientů. Lze jej nastavit i jako démona, který bude v nastavených intervalech stahovat poštu z nastavených serverů. Podrobnější informace a příklady použití získáte na informačních stránkách. V adresáři /usr/share/doc/fetchmail-<verze> naleznete úplný seznam všech funkcí i návod pro začátečníky.

Pomocí filtru procmail je možné příchozí poštu filtrovat, vytvářet konference, předzpracovávat zprávy, selektivně je předávat a podobně. Související program formail umožňuje mimo jiné gene-rovat automatické odpovědi a rozdělovat poštovní schránky. Procmail je na unixových a linuxo-vých systémech k dispozici již řadu let a jde o velmi robustní systém, navržený pro práci i za nej-horších okolností. Další informace naleznete v adresáři /usr/share/doc/procmail-<verze> a na manuálových stránkách.

Webové služby Webový server Apache

Apache je jednoznačně nejrozšířenější webový server; tento software používá více než polovina webových serverů na celém Internetu. Apache najdete ve většině linuxových distribucí. Mezi jeho hlavní výhody patří modulární design, podpora SSL, stabilita a rychlost. Při správné konfiguraci na vhodném hardwaru obstojí i v největším zatížení.

Na linuxových systémech se konfigurace tohoto serveru obvykle nastavuje v adresáři /etc/httpd. Nejdůležitějším konfiguračním souborem je httpd.conf; jeho obsah je velmi dobře komentován. Pokud byste potřebovali další informace, naleznete je jednak na manuálové stránce httpd a jed-nak v obsáhlé dokumentaci na adrese *http://www.apache.org/.*

Webové prohlížeče

Pro linuxovou platformu existuje celá řada webových prohlížečů, jak volně šířených, tak komerčních. Po dlouhou dobu byl jedinou použitelnou volbou prohlížeč Netscape Navigator, se vznikem projektu Mozilla je k dispozici důstojná konkurence. Mozilla Firefox v aktuální verzi je robustní a použitelný prohlížeč s množstvím rozšiřujících pluginů. Jeho renderovací jádro používá několik dalších linuxových prohlížečů, jako např. Epiphany nebo Galeon.
Dalším prohlížečem je Amaya, pocházející přímo od W3C. Rychlý a kompaktní je komerční pro-hlížeč Opera (existuje i ve verzi „zadarmo pro nekomerční použití“). Řada správců plochy obsa-huje funkci webového prohlížeče přímo ve správci souborů, jakým je například nautilus nebo konqueror.

Mezi oblíbené textové prohlížeče patří lynx a links. Prostřednictvím příslušných proměnných shellu můžete pro tyto prohlížeče nastavit i použití proxy serveru. Textové prohlížeče jsou rychlé a vhodné v případech, kdy není k dispozici grafické prostředí, například ve skriptech. Dalšími vynikajícími nástroji jsou curl a wget, použitelné ve skriptech, k rekurzivnímu stažení celého obsahu serveru a k hromadě dalších činností.

File Transfer Protocol FTP servery

Jako FTP server se na linuxových systémech velmi často používá *WU-ftpd*, spouštěný prostřed-nictvím xinetd. V případě hodně zatížených serverů jej však lze spustit i jako samostatný server.
Dalšími oblíbenými FTP servery jsou *Ncftpd* a *Proftpd*.Většina linuxových distribucí obsahuje také balík *anonftp*, který umožňuje snadno nastavit ano-nymní FTP server.

FTP klienti

Většina linuxových distribucí obsahuje program ncftp, což je vylepšená verze klasického unixo-vého příkazu ftp, který můžete znát i z příkazového řádku Windows. Program ncftp nabízí roz-šířené funkce, jako například pěknější a přehlednější uživatelské rozhraní, doplňování názvů sou-borů, funkce append a resume, záložky, správu relací a další:

thomas:~> ncftp blobNcFTP 3.0.3 (April 15, 2001) by Mike Gleason (ncftp@ncftp.com).Connecting to blob...blob.some.net FTP server (Version wu-2.6.1-20) ready.Logging in...Guest login ok, access restrictions apply.Logged in to blob.ncftp / > helpCommands may be abbreviated. 'help showall' shows hidden and unsupported commands. 'help <command>' gives a brief description of <command>.

ascii	cat	help	lpage	open	quote	site
bgget	cd	jobs	lpwd	page	rename	type
bgput	chmod	lcd	lrename	pdir	rhelp	umask
bgstart	close	lchmod	lrm	pls	rm	version
binary	debug	lls	lrmdir	put	rmdir	
bookmark	dir	lmkdir	ls	pwd	set	
bookmarks	get	lookup	mkdir	quit	show	

ncftp / >

Vynikající nápovědu s celou řadou příkladů naleznete na manuálových stránkách. I v tomto pří-padě je k dispozici také celá řada grafických klientů.

> Protokol FTP není bezpečný!
>
> Pokud přesně nevíte, co a proč děláte, nepoužívejte protokol FTP jinak než pro anonym-ní přístup. Případný útočník může být schopen odposlechnout vaše jméno a heslo. Je-li to možné, použijte zabezpečenou verzi protokolu FTP; program sftp je součástí balíku Secu-re Shell, viz kapitolu „Rodina SSH“.

Chat a konference

V jednotlivých distribucích jsou k dispozici různí klienti a různé systémy. Krátký a neúplný seznam nejoblíbenějších programů vypadá takto:

gaim: multiprotokolový klient pro Linux, Windows a Mac, kompatibilní se službami MSN, ICQ, IRC a s celou řadou dalších. Více informací naleznete na informačních stránkách a také přímo na domovských stránkách projektu, *http://gaim.sourceforge.net/*.
xchat: IRC klient pro systém X Window:

Domovskou stránku projektu najdete na adrese *http://sourceforge.net/projects/xchat/*.

JMSN: Klon Java MSN Messengeru s celou řadou vylepšení oproti originálu.
Konversation, KVIrc a řada dalších K- programů z rodiny KDE.
Ekiga (původně gnomemeeting): videokonferenční program pro Unix.
jabber: platforma s otevřeným kódem, podporující služby jako ICQ, AIM, Yahoo, MSN, IRC, SMTP a celou řadu dalších.
psi: jabber klient, viz *http://psi.affinix.com/*.
skype: proprietární program pro internetovou telefonii mezi uživateli služby Skype, další informace naleznete na adrese *http://www.skype.com*. Program je k dispozici zdarma, celý protokol je však uzavřený.
TeamSpeak: aplikace určená pro konferenční hovory, původně určená pro použití při počítačových hrách, více viz *http://www.goteamspeak.com/index.php*.

Služby news

Provozovat usenetový server vyžaduje řadu zkušeností a dolaďování, více informací naleznete na stránkách *http://www.isc.org*. V hierarchii *comp.** najdete několik zajímavých diskusních skupin, přistupovat k nim můžete pro-střednictvím celé řady textových i grafických klientů. Řada poštovních klientů podporuje i službu news, vyzkoušejte, zda to dokáže i vámi používaný klient. Můžete také nainstalovat některý z tex-tových klientů, jako jsou tin, slrnn nebo mutt, případně si rovnou stáhnout Thunderbird nebo jiný grafický klient.

Na adrese *http://groups.google.com/* naleznete archiv konferencí s možností vyhledávání. Jde

o vynikající místo, kde můžete hledat pomoc s různými problémy – je poměrně pravděpodobné, že na stejný problém už někdo narazil, vyřešil jej a řešení popsal v některé konferenci.

Domain Name System

Všechny dříve popsané služby využívají k překladu názvů na IP adresy a zpět služeb DNS. Server DNS samozřejmě nezná všechny IP adresy na celém světě, je ale schopen komunikovat s jinými servery DNS a potřebné informace zjistit. Na většině unixových systémů se používá server named, což je součást balíku BIND (Berkeley Internet Name Domain) distribuovaného konsorciem ISC. Server může pracovat i v režimu samostatného cache serveru, což je režim velmi často používaný ke zrychlení přístupu k síti.

Hlavním klientským konfiguračním souborem je /etc/resolv.conf, který definuje, s jakými servery DNS má klient komunikovat:

search somewhere.org nameserver 192.168.42.1 nameserver
193.74.208.137

Další informace můžete nalézt na informačních stránkách programu named, v souboru /usr/share/doc/bind-<verze> a na stránkách projektu BIND, *http://www.isc.org/index.pl?/ sw/bind/*. Použití balíku BIND popisuje dokument DNS HOWTO, *http://www.tldp.org/ HOWTO/DNS-HOWTO.html*.

DHCP

DHCP je Dynamic Host Configuration Protocol, který postupně vytlačuje starší používaný proto-kol bootp. Slouží k nastavení základních síťových parametrů, jako jsou IP adresy a názvy počíta-čů. Služba DHCP je zpětně kompatibilní s protokolem bootp. Budete-li server nastavovat, dopo-ručujeme vám přečíst si nejprve související dokumenty HOWTO.

Klientské počítače se typicky nastavují nějakým grafickým konfiguračním nástrojem (YaST2, ovlá-dací centrum Mandriva), který spouští dhcpcd, klientského démona služby DHCP. Budete-li chtít svůj počítač nastavit jako klienta služby DHCP, doporučujeme vám pročíst dokumentaci k systé-mu.

Autentizační služby Tradiční

Tradičně se autentizace uživatelů provádí lokálně prostřednictvím informací uložených v soubo-rech /etc/passwd a /etc/shadow. I pokud budete používat nějakou síťovou autentizační službu, tyto lokální soubory budou i nadále zapotřebí kvůli nastavení systémových a administrativních účtů, jako je účet root, účty různých démonů a další pomocné účty.
Tyto soubory jsou velmi často prvním cílem každého útočníka, ověřte proto, že mají správně nastavena přístupová práva a vlastnictví. Mělo by to vypadat takto:

bob:~> ls -l /etc/passwd /etc/shadow -rw-r--r--1 root root 1803 Mar 10 13:08 /etc/passwd -r--------1 root root 1116 Mar 10 13:08 /etc/shadow

PAM

V Linuxu se obvykle používá unixový autentizační mechanismus PAM. Mezi jeho výhody patří:

Společné autentizační schéma, které je možno používat v celé řadě aplikací.
PAM je možno použít s celou řadou aplikací bez nutnosti tyto aplikace specificky překládat.
Administrátoři a tvůrci programů mají k dispozici velmi pružný a přizpůsobivý autentizační mechanismus.
Autor aplikace nepotřebuje v aplikaci implementovat konkrétní autentizační metody, může se soustředit čistě na funkčnost aplikace.

V adresáři /etc/pam.d naleznete konfigurační soubory mechanismu PAM (dříve se používal jediný soubor /etc/pam.conf). Každá aplikace nebo služba zde má vlastní soubor. Každý řádek takového souboru obsahuje čtyři údaje:

- *Modul:*

auth: Zajišťuje vlastní autentizaci (například vyzve k zadání hesla a ověří je) a na základě toho provádí další operace, například nastavuje členství ve skupinách nebo generuje lístky pro Kerberos.
account: Ověřuje, že pro daného uživatele je přístup povolen (nevypršela platnost účtu, uživatel má povoleno přihlášení v daném čase a podobně).
password: Slouží na nastavování hesel.
session: Používá se po ověření uživatele. Tento modul provádí další úkony nutné k zajištění přístupu (například připojení domovského adresáře uživatele nebo zpřístupnění jeho poštovní schránky).

Velmi důležité je pořadí, v jakém jsou moduly používány jeden po druhém.

Řídící příznaky: Říkají mechanismu PAM, co se má provést v případě úspěšné či neúspěš-né autentizace. Možné hodnoty jsou

required, requisite, sufficient a optional.
Cesta k modulu: cesta k používanému zásuvnému modulu, typicky jsou uloženy v adresá-ři /lib/security.
Parametry: další informace pro modul.

PAM automaticky detekuje používání stínových hesel.Další informace naleznete na manuálové stránce pam a na webových stránkách projektu Linux-PAM na adrese *http://www.kernel.org/pub/linux/libs/pam/.*

LDAP

Protokol Lightweight Directory Access Protocol je systém klient/server určený pro přístup ke glo-bálním nebo lokálním adresářovým službám přes síť. V Linuxu se používá implementace Open-LDAP. Ta zahrnuje programy slapd, samostatný server, slurpd, samostatný replikační server, kni-hovny implementující protokol LDAP a sadu utilit, nástrojů a vzorových klientů.
Hlavní výhoda při použití LDAP spočívá v konsolidaci některých typů informací v rámci organi-zace. Do jednoho adresáře je možné například sloučit všechny používané seznamy uživatelů. S adresářem pak může pracovat jakákoliv aplikace podporující LDAP, která dané informace potře-buje. S adresářem mohou pracovat rovněž přímo individuální uživatelé.

Mezi další výhody LDAP patří poměrně snadná implementace (v porovnání se standardem X.500) a dobře definované aplikační rozhraní, díky němuž stále roste počet aplikací a bran, které LDAP podporují.

Nevýhodou je, že chcete-li LDAP používat, potřebujete buď aplikace podporující LDAP nebo musí-te mít možnost používat LDAP brány. I když obliba LDAP stále roste, v současné době neobsa-huje Linux příliš mnoho aplikací, které tento protokol podporují. Další nevýhodou je, že i když LDAP obsahuje nějaké mechanismy pro řízení přístupu, jeho bezpečnostní funkce nejsou tak pro-pracované jako u X.500.

LDAP je otevřený protokol s širokými možnostmi nastavení, lze jej použít k uložení prakticky všech typů informací, které se týkají konkrétní organizační struktury. Mezi typické aplikace patří uložení e-mailových adres, centrální autentizace ve spolupráci s PAM, seznamy telefonních čísel a databáze konfigurací počítačů.

Konkrétní informace k souvisejícím programům, jako jsou ldapmodify nebo ldapsearch, najde-te na manuálových stránkách a v dokumentaci k systému. Další informace obsahuje dokument LDAP Linux HOWTO, *http://www.tldp.org/HOWTO/LDAP-HOWTO/,* který popisuje instalaci, kon-figuraci, spuštění a údržbu LDAP serveru na Linuxu. Dokument LDAP Implementation HOWTO, *http://www.tldp.org/HOWTO/LDAP-Implementation-HOWTO/,* popisuje technické podrobnosti týkající se ukládání aplikačních dat na serveru LDAP. Autor této příručky je rovněž autorem doku-mentu LDAP Operations HOWTO, *http://tille.xalasys.com/training/ldap/,* který obsahuje základní informace týkající se správy a užívání LDAP a integrace služeb.

Vzdálené spouštění aplikací

Úvod

Existuje několik různých způsobů, jak na vzdáleném počítači provést příkaz nebo spustit program a jeho textový či grafický výstup sledovat na místním počítači. Spojení mezi počítači může být zabezpečené nebo nezabezpečené. I když je samozřejmě vhodné používat zabezpečené metody komunikace a nepřenášet hesla po síti nešifrovaně, budeme hovořit i o některých starších (neza-bezpečených) mechanismech, které jsou stále užitečné i v moderním síťovém prostředí, ať už pro účely diagnostiky nebo ke spouštění exotických programů.

Rsh, rlogin a telnet

Příkazy rlogin a rsh pro vzdálené přihlašování a vzdálené spouštění příkazů pocházejí z Unixu. Dnes už se používají jen velmi zřídka vzhledem k jejich katastrofickým bezpečnostním vlastnos-tem, nicméně jsou stále součástí prakticky každé linuxové distribuce kvůli zpětné kompatibilitě s unixovými programy.

Na druhé straně stojí dodnes velmi často používaný program telnet, užívaný jak systémem, tak i správci systémů. Telnet je jeden z nástrojů pro vzdálený přístup k souborům a pro vzdálenou administraci, umožňuje připojení odkudkoliv z Internetu. V kombinaci s X serverem je možné vzdálené grafické aplikace zobrazovat lokálně. Mezi prací na lokálním počítači a na vzdáleném počítači tak není žádný rozdíl.
Celé spojení je ovšem nešifrované, takže povolujete-li připojení příkazem telnet, je to značné bez-pečnostní riziko. K běžnému spouštění vzdálených programů se doporučuje Secure Shell nebolissh. O tomto programu budeme hovořit zanedlouho.
I tak je ale telnet v mnoha případech velmi užitečný. Následující příklady ukazují, jak jím lze ově-řit funkčnost poštovního serveru a webového serveru.Kontrola, zda funguje poštovní server:

```
[jimmy@blob ~] telnet mailserver 25 Trying 192.168.42.1... Connected to mailserver. Escape character is '^]'. 220 m1.some.net ESMTP Sendmail 8.11.6/8.11.6; 200302281626 ehlo some.net 250-m1.some.net Hello blob.some.net [10.0.0.1], pleased to meet you 250-ENHANCEDSTATUSCODES 250-8BITMIME 250-SIZE 250-DSN 250-ONEX 250-ETRN 250-XUSR 250 HELP
mail from: jimmy@some.net
250 2.1.0 jimmy@some.net... Sender ok
```

```
rcpt to: davy@some.net
250 2.1.5 davy@some.net... Recipient ok
data
354 Enter mail, end with "." on a line by itself
test .
250 2.0.0 g2MA1R619237 Message accepted for delivery
quit
221 2.0.0 m1.some.net closing connection Connection closed by foreign host.
```

Kontrola, zda webový server reaguje na základní požadavky:

```
[jimmy@blob ~] telnet www.some.net 80Trying 64.39.151.23...Connected to www.some.net.Escape character is '^]'.
HEAD / ;HTTP/1.1

HTTP/1.1 200 OK Date: Fri, 22 Mar 2002 10:05:14 GMT Server: Apache/1.3.22 (UNIX) (Red-Hat/Linux)
  mod_ssl/2.8.5 OpenSSL/0.9.6
  DAV/1.0.2 PHP/4.0.6 mod_perl/1.24_01 Last-Modified: Fri, 04 Jan 2002 08:21:00 GMT ETag: "70061-68-3c3565ec" Accept-Ranges: bytes
Content-Length: 104 Connection: close Content-Type: text/html Connection closed by foreign host.

[jimmy@blob ~]
```

Takovéto použití je zcela bezpečné, protože nikde nezadáváte žádné jméno a heslo a nikdo tak není schopen tyto citlivé údaje odposlechnout.

Systém X Window Funkce X serveru

Jak už jsme říkali v kapitole „Home, sweet /home“ (viz kapitolu „Konfigurace X serveru“), systém X Window obsahuje X server, poskytující grafické operace klientům, kteří chtějí něco zobrazit. Je důležité rozlišovat mezi X serverem a klientskými X aplikacemi. X server přímo obsluhuje zobrazování a zodpovídá za obsluhu veškerých vstupů přes klávesnici, myš i displej. Na druhé straně X klient nemá přímý přístup ke vstupním a výstupním zařízením. Místo toho komunikuje s X serverem, který vstupy a výstupy obsluhuje. X klient dělá tu „opravdovou práci“, jako je výpo-čet hodnot, běh aplikace a podobně. X server jen otevře okno, které pro daného klienta obslu-huje jeho vstupy a výstupy.

V normálním režimu práce (úroveň běhu 5, grafický režim) funguje každá linuxová pracovní sta-nice jako X server sama pro sebe a běží na ní řada klientských aplikací. Všechny uživatelem spouš-těné aplikace (například GIMP, terminálové okno, prohlížeč, kancelářský balík, přehrávač CD a podobně) jsou klienty lokálního X serveru. V tomto případě server i klienti běží na stejném počí-tači.

Díky této architektuře je systém X Window ideálním prostředím pro vzdálené spouštění aplikací a programů. Proces samotný běží na vzdáleném počítači, takže lokální stanice má jen minimální nároky na procesor. Tyto systémy, fungující čistě jako X servery, se označují názvem X terminály a svého času byly velmi populární. Více informací k tomuto tématu najdete v dokumentu Remo-te X applications miniHOWTO, *http://www.tldp.org/HOWTO/mini/Remote-X-Apps.html.*

Telnet a X

Pokud byste chtěli prostřednictvím telnetu zobrazit grafickou aplikaci běžící na vzdáleném stroji, musíte nejprve tomuto vzdálenému systému povolit přístup na váš display (váš X server), a to pří-kazem xhost. V lokálním terminálovém okně zadáte přibližně následující příkaz:

```
davy:~> xhost +remote.machine.com
```

Následně se můžete připojit na vzdálený počítač a říct mu, aby grafický výstup aplikace zobrazo-val na vašem lokálním systému. K tomu slouží proměnná prostředí DISPLAY:

```
[davy@remote ~] export DISPLAY="local.host.com:0.0"
```

V takto nastaveném prostředí se bude každá aplikace spuštěná ve vzdáleném terminálovém okně zobrazovat na vašem lokálním systému, přičemž pro své interní výpočetní účely bude používat procesor vzdáleného systému, pro účely zobrazení pak vaše lokální prostředky.

Tento postup předpokládá, že na lokálním systému, na němž chcete aplikaci zobrazovat, už máte nějaký X server nastaven (například XFree86/X.org, Exceed, Cygwin). Architektura X serveru a operační systém klientského počítače nejsou podstatné.

Zobrazení terminálového okna ze vzdáleného systému je rovněž chápáno jako zobrazení grafiky.

Rodina SSH Úvod

Většina unixových a linuxových systémů dnes preferuje použití Secure Shellu, který eliminuje bezpeč-nostní rizika telnetu. Na linuxových systémech se většinou používá implementace OpenSSH, otevře-ná implementace protokolu SSH, která nabízí

zabezpečenou šifrovanou komunikaci mezi nedůvěry-hodnými počítači po nedůvěryhodné síti. Při standardním nastavení dochází k automatickému forwar-dování spojení na X server, je ale možno nastavit forwardování provozu na libovolných portech.

Klient ssh se připojuje a přihlašuje k zadanému vzdálenému počítači. Uživatel musí na vzdálenémsystému prokázat svou identitu, a to některou z metod definovaných v konfiguračním souborusshd_config, který se typicky nachází v adresáři /etc/ssh. Konfigurační soubor je velmi dobřeokomentován a ve výchozím nastavení zapíná většinu běžně používaných funkcí. Pokud bystepotřebovali další informace, naleznete je na manuálových stránkách sshd.

Jakmile server ověří identitu uživatele, tak buď provede požadovaný příkaz anebo se ke vzdále
ném u systému přihlásí a poskytne uživateli běžný shellový přístup. Veškerá komunikace se vzdá
leně spuštěným příkazem či shellem je automaticky šifrována.

Spojení končí ve chvíli, kdy skončí vzdáleně prováděný příkaz či shell a dojde k uzavření všech

navázaných X11 a TCP/IP spojení.Při prvním připojování ke vzdálenému systému kterýmkoliv z programů z rodiny SSH musítepotvrdit identitu vzdáleného systému a také to, že se k němu chcete připojit.

lenny ~> ssh blobThe authenticity of host 'blob (10.0.0.1)' can't be established.RSA fingerprint is 18:30:50:46:ac:98:3c:93:1a:56:35:09:8d:97:e3:1d.Are you sure you want to continue connecting (yes/no)? yesWarning: Permanently added 'blob,192.168.30.2' (RSA) to the list of known hosts.Last login: Sat Dec 28 13:29:19 2002 from octarineThis space for rent.

lenny is in ~

Důležité je odpovědět skutečně „yes“, nestačí pouze „y“. V tomto okamžiku dojde k editaci lokál

ního souboru ~/.ssh/known_hosts, viz kapitolu „Autentizace serveru“.Pokud chcete na vzdáleném systému jen něco ověřit a hned se vrátit zpět na lokální systém, může-te jako parametr příkazu ssh uvést příkaz, který chcete na vzdáleném systému provést:

lenny ~> ssh blob who jenny@blob's password:

root	tty2	Jul 24 07:19
lena	tty3	Jul 23 22:24
lena	0:	Jul 25 22:03

lenny ~> uname -n magrat.example.com

Přesměrování X11 a TCP spojení

Pokud je konfigurační direktiva X11Forwarding nastavena na hodnotu *yes* a uživatel spustí na vzdáleném systému nějakou X aplikaci, dojde k nastavení proměnné prostředí DISPLAY a spoje-ní na displej X serveru se automaticky přesměrovává na místní systém tak, že jakékoliv X progra-my spuštěné ze vzdáleného shellu budou s místním systémem komunikovat po zabezpečeném kanálu a k (nezabezpečenému) spojení na místní X server dojde až z ssh klienta na místním syste-mu. Uživatel ručně nenastavuje proměnnou DISPLAY. Předávání spojení protokolem X11 lze nastavit buď z příkazového řádku nebo v konfiguračním souboru sshd.

Hodnota proměnné DISPLAY nastavená příkazem ssh bude ukazovat na vzdálený systém, číslo displeje ale bude větší než nula. To je normální chování a dochází k němu proto, že ssh vytvoří na vzdáleném systému (na němž běží X klient) *X proxy server*, který bude spojení zabezpečeným kanálem předávat na X server na místním systému.

Toto všechno se odehrává automaticky, takže jakmile spustíte grafickou aplikaci, zobrazí se na vašem místním počítači, a nikoliv na vzdáleném systému. Jako příklad uvádíme program xclock, protože je malý, obecně rozšířený, a je tak velmi vhodný pro testovací účely:

SSH na serveru automaticky nastaví hodnotu Xauthority. Za tím účelem vygeneruje náhodné autorizační cookie, uloží jej na serveru v Xauthority, ověřuje, že všechna přesměrovaná spojení toto cookie obsahují, a po otevření spojení je nahrazuje hodnotou skutečného autorizačního cookie. Skutečné autorizační cookie se tak na server nikdy nepřenáší (a vůbec nic se nepřenáší nešifrovaně).

Na příkazovém řádku nebo v konfiguračním souboru je možné nastavit forwardování TCP/IP spojení na libovolném portu.

X server

Výše popsaná procedura předpokládá, že na místním počítači, na němž chcete zobrazovat vzdáleně spouštěné aplikace, máte nastaven funkční X server. Architektura a operační systém místního a vzdáleného počítače se mohou lišit, podmínkou je pouze bežící X ser-ver – například Cygwin (což je implementace serveru XFree86 pro MS Windows) nebo

Exceed. Vzdálené spojení mezi libovolnými linuxovými nebo unixovými systémy by mělo být možno navázat bez jakýchkoliv problémů.

Autentizace serveru

Program ssh si udržuje a kontroluje databázi s identifikacemi všech serverů, k nimž se připojoval. Klíče serverů se ukládají v domovském adresáři uživatele v souboru $HOME/.ssh/known_hosts. Kromě toho se servery vyhledávají také v souboru /etc/ssh/ssh_known_hosts. Při prvním připoje-ní ke každému serveru dojde k jeho přidání do uživatelské databáze známých serverů. Pokud se identifikátor serveru někdy v budoucnu změní, ssh na to upozorní a zároveň zakáže autentizaci heslem, aby eventuální podvržený server nemohl zjistit heslo uživatele. Dalším důvodem pro pou-žití tohoto mechanismu je ochrana před útoky typu man-in-the-middle, jimiž by bylo jinak možné ochranu šifrováním překonat. V prostředích, kde jsou vysoké nároky na bezpečnost, je možné ssh nastavit tak, aby vůbec nebylo možné připojit se k serverům s neznámým nebo změněným klíčem.

Bezpečné vzdálené kopírování

Součástí rodiny SSH je program scp, který představuje bezpečnou náhradu programu rcp, použí-vaného v dobách, kdy existovalo jen rsh. Program scp používá k přenosu dat ssh, používá stejné autentizační mechanismy a poskytuje stejnou míru zabezpečení. Na rozdíl od rcp se scp v přípa-dě potřeby zeptá na heslo nebo tzv. passfrázi (heslo k ssh klíči, viz dále), potřebné k autentizaci:

```
lenny /var/tmp> scp Schedule.sdc.gz blob:/var/tmp/lenny@blob's password:Schedule.sdc.gz 100% |*****************************| 100 KB 00:00

lenny /var/tmp>
```

Součástí názvu zdrojového nebo cílového souboru může být uživatelské jméno a název vzdále-ného systému, čímž se říká, že se soubor kopíruje z nebo na tento systém. Přímé kopírování mezi dvěma vzdálenými systémy není možné. Další podrobnosti naleznete na informačních stránkách.

Pokud raději používáte rozhraní podobné programy FTP, použijte program sftp:

```
lenny /var/tmp> sftp blob Connecting to blob... lenny@blob's
password:

sftp> cd /var/tmp


sftp> get Sch*Fetching /var/tmp/Schedule.sdc.gz to Schedule.sdc.gz

sftp> bye

lenny /var/tmp>
```

Bezpečné kopírování nebo grafické FTP

Nevyhovuje vám práce v příkazovém řádku? Pak vyzkoušejte Konqueror a jeho možnosti bezpečného kopírování anebo si nainstalujte Putty.

Autentizační klíče

Příkaz ssh-keygen generuje, spravuje a konvertuje autentizační klíče používané programem ssh. Dokáže generovat RSA klíče používané SSH protokolem verze 1 a RSA nebo DSA klíče používa-né protokolem verze 2.
Normální uživatel, který bude chtít SSH používat společně s RSA či DSA autentizací, si jednou vygeneruje klíče, které se uloží do souboru $HOME/.ssh/identity, id_dsa nebo id_rsa. Správce systému tímto programem může vygenerovat klíče počítače.

Program vygeneruje klíč a zeptá se na název souboru, do nějž má uložit privátní část klíče. Veřej-ná část klíče bude uložena ve stejnojmenném souboru, navíc s příponou *.pub*. Kromě toho se pro-gram zeptá na passfrázi. Ta může být buď prázdná (klíče počítače musí mít prázdnou passfrázi) anebo to může být libovolně dlouhý řetězec. Tento řetězec pak zadáváte při každém použití dané-ho klíče, ovšem pro lepší správy klíčů (a hesel) můžete využít ssh-agent.

Zapomenutou passfrázi není možné nijak obnovit. Pokud ji ztratíte nebo zapomenete, musíte vygenerovat nové klíče.
Problematice SSH klíčů se budeme věnovat ve cvičeních. Všechny potřebné informace naleznete na manuálových a informačních stránkách.

VNC

VNC či Virtual Network Computing je fakticky systém vzdáleného displeje, který umožňuje zobra-zit prostředí pracovní plochy nejen na lokálním počítači, na němž se plocha „nachází“, ale sou-časně i na jakémkoliv jiném počítači na Internetu, a to na celé řadě systémů a architektur včetně MS Windows a několika unixových distribucí. Můžete mít například ve Windows NT spuštěný MS Word a plochu zobrazovat na linuxové stanici. VNC obsahuje servery i klienty, takže je možné i opačné uspořádání – na stanicích s Windows je možno zobrazovat linuxové pracovní prostředí. VNC představuje pravděpodobně nejjednodušší metodu pro nastavení vzdáleného grafického pří-stupu. Od klasického X řešení a různých komerčních implementací se VNC liší zejména v těchto vlastnostech:

Na straně „prohlížeče“ nejsou uloženy žádné stavové údaje, takže můžete vzdálenou plo-chu opustit, připojit se na ni z jiného počítače a pokračovat v práci. Pokud na počítači pro-vozujete X server a počítač zhavaruje, všechny vzdálené aplikace zobrazované tímto ser-verem budou ukončeny. Při použití VNC se s aplikacemi nic nestane.
Program je malý a jednoduchý, nevyžaduje instalaci, v případě potřeby jej lze spustit z diskety.
Klient v Javě zajišťuje platformní nezávislost, můžete běžet prakticky na jakémkoliv systé-mu s podporou grafiky.
Možnost sdílení – stejnou plochu je možné současně zobrazit ve více prohlížečích.
Licence GPL.

Více informací naleznete na manuálové stránce man vncviewer nebo na webových stránkách *http://www.realvnc.com/*.

Protokol rdesktop

Kvůli usnadnění administrace počítačů se systémem MS Windows obsahují novější linuxové distri-buce podporu protokolu RDP (Remote Desktop Protocol), která je implementována v klientu rdesktop. Tento protokol se používá v řadě produktů společnosti Microsoft, například ve Win-dows NT Terminal Server, Windows 2000 Server, Windows XP nebo Windows 2003 Server.

Můžete své kolegy (nebo management) překvapit celoobrazovkovým režimem s podporou více rozložení klávesnic úplně stejně jako v originálním programu. Více informací naleznete na manu-álové stránce man rdesktop nebo na webových stránkách *http://www.rdesktop.org/*.

Bezpečnost

Úvod

Jakmile počítač připojíte k síti, vzniká riziko mnoha různých způsobů jeho zneužití, ať už jde o unixový nebo jakýkoliv jiný systém. Na toto téma byly popsány celé stohy papíru a v tomto dokumentu nemáme prostor pro detailní debatu na téma počítačové bezpečnosti. I úplný začá-tečník se ale může řídit několika striktně logickými pravidly, při jejichž dodržování získá vysoce bezpečný systém. Většina úspěšných útoků má totiž příčinu v neznalosti nebo bezstarostnosti uži-vatele.

Možná se ptáte, zda se vás toto téma vůbec týká, když počítač používáte doma anebo v práci v chráněném prostředí. Raději byste si ale měli klást následující otázky:

Chcete, aby váš počítač sledoval vaše aktivity?
Chcete se (nevědomky) podílet na nezákonných aktivitách?
Chcete, aby vaše zařízení používal někdo jiný?
Chcete přijít o své připojení k Internetu?
Chce se vám přeinstalovávat počítač pokaždé, když jej někdo úspěšně napadne?
Chcete riskovat ztrátu osobních či jiných dat?

Předpokládáme, že po ničem z uvedeného netoužíte, proto si představíme krátký seznam kroků, jimiž můžete svůj počítač zabezpečit. Podrobnější informace naleznete například v dokumentu Linux Security HOWTO na adrese *http://www.tldp.org/HOWTO/Security-HOWTO/*.

Služby

Vtip je v tom, provozovat co nejméně možných služeb. Čím méně portů máte otevřeno do vněj-šího světa, tím lépe. Pokud některé služby na lokální síti nemůžete vypnout, zabraňte alespoň v přístupu k těmto službám zvenčí.

Základní pravidlo říká, že pokud nějakou službu neznáte, nejspíš ji stejně nepotřebujete. Některé služby navíc nejsou určeny k použití přes Internet. Nespoléhejte na to, co by běžet *mělo*, zkon-trolujte seznam otevřených portů pomocí příkazu netstat:

[elly@mars ~] netstat -l | grep tcp tcp 0 0 *:32769 *:* LISTEN tcp 0 0 *:32771 *:* LISTEN tcp 0 0 *:printer *:* LISTEN tcp 0 0

*:kerberos_master *:* LISTEN tcp 0 0 *:sunrpc *:* LISTEN tcp 0 0 *:6001 *:* LISTEN tcp 0 0 *:785 *:* LISTEN tcp 0 0
localhost.localdom:smtp *:* LISTEN tcp 0 0 *:ftp *:* LISTEN tcp 0 0 *:ssh *:* LISTEN tcp 0 0 ::1:x11-ssh-offset *:* LISTEN

Čemu byste se měli vyhnout:

exec, rlogin, rsh a telnet jako základní bezpečnostní pravidlo,
X Window na serverech,
lp, není-li připojena tiskárna,
nejsou-li v síti Windows, nepotřebujete Sambu,
pokud nutně nepotřebujete FTP, vyhněte se mu,
NFS a NIS nikdy nepovolujte přes Internet, pokud tyto služby nepotřebujete, nezapínejte je vůbec,
nespouštějte MTA, pokud váš počítač není poštovním serverem,
atd.

Seznam spouštěných služeb můžete měnit příkazem chkconfig, editací inicializačních skriptů a konfiguračních souborů služby (x)inetd. Lze použít i distribuční nástroje jako ntsysv, YaST2, DrakXServices apod.

Pravidelně systém aktualizujte

Úspěch Linuxu spočívá v jeho schopnosti rychle se adaptovat na stále se měnící podmínky. Záro-veň tak ale může docházet k situacím, kdy je vydána bezpečnostní záplata už ve chvíli, kdy insta-lujete úplně novou verzi operačního systému. Proto byste měli vždy bezprostředně po dokonče-ní instalace (a to platí pro jakýkoliv OS) ihned provést aktualizaci systému. I poté musíte systém pravidelně aktualizovat.

Při některých aktualizacích dojde k vytvoření nových konfiguračních souborů a mohou být pře-psány staré. Ověřte si to v dokumentaci a po každé aktualizace zkontrolujte, že všechno funguje správně.

Většina linuxových distribucí nabízí možnost přihlášení do konference, v níž se oznamují vydáva-né bezpečnostní aktualizace, a většina také obsahuje nástroje pro aktualizaci systému. Obecné bezpečnostní záležitosti se mimo jiné probírají i na *http://www.linuxsecurity.com/*.

Aktualizace nikdy nekončí, proto byste je měli provádět téměř denně.

Firewally a přístupové politiky Co je to firewall?

V předchozím textu jsme se už zmínili, že Linux dokáže fungovat i jako firewall. Správa firewallu je jedním z úkolů správce sítě, i běžný uživatel by však měl mít o této problematice určité povědomí.

Firewall je poměrně vágní termín, jímž se označuje cokoliv, co stojí jako ochranná bariéra mezi námi a vnějším světem, obecně Internetem. Firewall může být realizován jako vyhrazený systém anebo může jít o specifickou aplikaci, která zajišťuje příslušnou funkčnost. Může také jít o kom-binaci komponent, včetně různých kombinací hardwaru a softwaru. Firewally jsou tvořeny „pravidly“, která definují, co může a nemůže do daného systému či sítě vstoupit, případně je opustit.

Po vypnutí všech nepotřebných služeb následuje omezení přístupu ke spuštěným službám na co nejmenší potřebnou množinu spojení. Příkladem může být práce z domu – v takovém případě mají být povolena jen konkrétní spojení mezi vaším domácím počítačem a pracovním počítačem, připojení z jiných počítačů na Internetu má být zakázáno.

Paketové filtry

První obrannou linii představuje *paketový filtr*, který zkoumá obsah IP paketů a na jeho základě se rozhoduje. Na starých jádrech 2.2 byl k dispozici paketový filtr ipchains. Nové systémy (jádra
2.4 a vyšší) používají iptables, novou generaci paketového filtru. K dispozici jsou i GUI nástroje, například Firestarter nebo Gnome nástroj Lokkit. Tyto nástroje tvoří pouze rozhraní, které má usnadnit práci běžným uživatelům. Dokáže nastavit základní chování firewallu pro běžnou pra-
covní stanici připojenou vytáčeným připojením či kabelovým modemem. Ve větších prostředích

jej nelze použít.Jedním z nejvýznamnějších vylepšení moderních jader je *stavová inspekce*, která zjišťuje, nejenomco paket obsahuje, ale i zda se paket vztahuje k již existujícímu spojení, nebo jde o novou komu-nikaci. Vývoj stále probíhá, takže je rozumné s každou novou verzí distribuce zjistit, co je nového.
Další informace naleznete na stránkách projektu netfilter/iptables na adrese

http://www.netfilter.org/.

TCP wrappery

TCP wrapper dává výsledky velmi podobné paketovému filtru, pracuje ale na jiném principu. Wrapper přijme pokus o spojení a teprve poté na základě svého konfiguračního souboru roz-hodne, jestli požadavek přijme nebo odmítne. Kontrola tak probíhá na

aplikační úrovni, nikoliv na síťové úrovni.

TCP wrappery se typicky používají ve spolupráci s xinetd, kdy zajišťují řízení přístupu na bázi IP adres. Navíc tyto nástroje nabízejí i funkce pro logování a správu zatížení, které se velmi snadno nastavují.

Výhodou TCP wrapperů je, že připojující se klient o přítomnosti wrapperu nic neví a že wrapper funguje nezávisle na aplikaci, kterou chrání. Řízení přístupu podle IP adres se nastavuje v souborech hosts.allow a hosts.deny. Další informa-ce můžete nalézt jednak v dokumentaci k wrapperu v souboru /usr/share/doc/tcp_wrappers-<verze>/, jednak na manuálových stránkách uvedených konfiguračních souborů, kde najdete i pří-klady jejich nastavení.

Proxy servery

Proxy servery mohou plnit různé funkce, ne všechny nutně souvisejí s bezpečností. Nicméně sku-tečnost, že proxy server vždy představuje mezilehlou vrstvu, z něj dělá vhodné místo, kde je možné vynutit přístupové politiky, omezit počet spojení na firewall a určit, jak se síť za proxy ser-verem bude tvářit zvenčí.

V kombinaci s paketovými filtry, v některých případech i samostatně, slouží proxy servery jako další vrstva pro řízení přístupu. Více informací můžete nalézt v dokumentu Firewall HOWTO na adrese *http://www.tldp.org/HOWTO/Firewall-HOWTO.html,* případně na webových stránkách pro-gramu Squid na adrese *http://www.squid-cache.org/.*

Přístup k jednotlivým aplikacím

Některé servery mají implementovány vlastní funkce pro řízení přístupu. Typickými příklady jsou Samba, X11, Bind, Apache a CUPS. U každé používané služby si ověřte, jakými konfiguračními soubory se nastavuje.

Logy

Unixový zvyk zaznamenávat v různých souborech různé aktivity lze v případě jakýchkoliv pro-blémů použít k ověření, „že to něco dělá“. Samozřejmě byste tyto soubory měli kontrolovat pra-videlně, ať už ručně nebo automaticky. Firewally a další podobné systémy pro řízení přístupu mají tendenci vytvářet velmi objemné logy, snahou proto je vhodně zachytit pouze neobvyklé aktivity.

Detekce průniku

Systémy pro detekci průniků (IDS, Intrusion Detection System) jsou navrženy tak, aby dokázaly zachytit to, co prošlo firewallem. Obvykle buď reagují na aktivně probíhající pokus o průnik nebo dokážou proběhnuvší průnik detekovat zpětně. Ve druhém případě už je sice pozdě na to, zabrá-nit škodám, přinejmenším se ale o problému dozvíte. Existují dva základní typy IDS: ty, které chrá-ní celou síť, a ty, které chrání jen jednotlivé počítače.

U počítačově zaměřených IDS se používají nástroje, které sledují změny v souborovém systému. Dojde-li ke změně systémového souboru, který by neměl být změněn, je to jasným důkazem toho, že se stalo něco zlého. Jakmile někdo získá k nějakému systému přístup, typicky se pokusí v systé-mu něco změnit. Obvykle k tomu dojde velmi rychle, cílem těchto změn je buď vytvoření zadních vrátek do budoucna anebo rovnou zahájení útoku na další systém. V obou případech ovšem dojde ke změnám souborů, nebo i k přidání nových souborů. Některé systémy obsahují sledovací systém tripwire, jehož popis naleznete na webových stránkách *http://www.tripwire.com/.*

Síťově zaměřené IDS jsou realizovány nějakým systémem, který vidí veškerý provoz procházející firewallem. Příkladem takového programu je Snort, *http://www.snort.org/.*

Další doporučení

Několik obecných doporučení, na která byste neměli zapomínat:

Nepovolujte přihlášení uživatele *root*. Více než 20 let existuje příkaz su, kterým zvýšíte zabezpečení systému.
Berte vážně hesla. Používejte stínová hesla a měňte je pravidelně.
Vždy se snažte používat SSH. Vyhýbejte se programům telnet, službě FTP a dalším kli-entským programů, jako jsou například poštovní klienti POP3, kteří po síti posílají nešifro-vaná hesla.
Omezte užívání prostředků pomocí příkazů quota a/nebo ulimit.
Pošta pro uživatele *root* by měla být doručována konkrétním osobám.
Institut SANS, *http://www.sans.org/,* nabízí celou řadu dalších tipů a triků rozdělených podle distribuce, provozují také poštovní konferenci. Doporučují mimo jiné použití zabez-pečovacího systému Bastille, *http://www.bastille-linux.org/.*
Ověřujte si původ nových programů, vždy programy stahujte z důvěryhodných míst. Před instalací kontrolujte integritu instalovaných balíků.
Nemáte-li trvalé připojení k Internetu, ukončete je vždy, jakmile je více nepotřebujete.
Privátní služby provozujte na netypických portech, vyhýbejte se známým číslům portů.
Poznejte svůj systém. Po nějaké době získáte cit pro to, že se v systému něco neobvyklého děje.

Byl systém napaden?

Jak to poznáte? Mezi podezřelé stavy patří:

Záhadné otevřené porty, podivné procesy.
Neobvykle se chovající systémové nástroje (běžné příkazy).
Problémy s přihlášením.
Neodůvodněné vytížení přenosové linky.
Poškozené nebo chybějící logy, podivně se chovající syslog.
Rozhraní v neobvyklém režimu.
Neočekávaně změněné konfigurační soubory.
Podivné položky v historii shellu.
Neidentifikované dočasné soubory.

Obnova po průniku

Hlavně zůstaňte klidní. V následujícím pořadí proveďte uvedené operace:

Odpojte počítač od sítě.
Zkuste co nejvíce zjistit o tom, jak došlo k průniku do systému.
Zálohujte důležitá nesystémová data. Je-li to možné, porovnejte data s existujícími záloha-mi z doby před napadením systému, abyste ověřili integritu dat.
Přeinstalujte systém.
Použijte nová hesla.
Obnovte data ze zálohy.
Nainstalujte všechny dostupné aktualizace.
Znovu systém prověřte – zablokujte nepotřebné služby, zkontrolujte pravidla firewallu a další přístupové politiky.
Znovu připojte počítač k síti.

Shrnutí

Linux a sítě spolu úzce souvisejí. Jádro Linuxu podporuje všechny běžné a celou řadu méně běž-ných síťových protokolů. V každé distribuci naleznete standardní unixové síťové nástroje. Kromě nich obsahuje většina distribucí nástroje pro snadnou instalaci a správu sítě.

Linux je známý svou stabilitou, pro niž je velmi vhodný k provozování různých síťových služeb. Různých programů je k dispozici nespočet. Stejně jako Unix je i Linux možno plnohodnotně pou-žívat a spravovat vzdáleně některou z mnoha metod pro vzdálené spouštění programů.
Stručně jsme se věnovali problematice bezpečnosti. Linux je ideální jako firewall, nenáročný a levný, lze jej však použít i v jiných funkcích, například jako směrovač nebo proxy server. Zabezpečení systému je primárně zajišťováno pravidelnou aktualizací a zdravým rozumem.

Cvičení

Sítě obecně

Vypište síťové informace o svém počítači: IP adresu, směrování, servery DNS.
Řekněme, že není k dispozici služba DNS. Jakým způsobem byste nastavili přístup ke kolegově počítači tak, abyste nemuseli vždy zadávat jeho IP adresu?
Jak trvale uložíte nastavení proxy u textových prohlížečů, jako je například links?
Které jmenné servery obsluhují doménu redhat.com?
Pošlete si poštu na lokální účet. Vyzkoušejte dva různé způsoby odeslání zprávy a jejího přečtení. Jak můžete ověřit, že vám dorazila nějaká zpráva?
Podporuje váš počítač připojení přes anonymní FTP? Jak byste se pomocí programu ncftp autentizovali svým jménem a heslem?
Běží na vašem počítači webový server? Pokud ne, nastavte jej. Zkontrolujte logy!

Vzdálené připojení

Zobrazte grafickou aplikaci, například xclock, spuštěnou na vašem počítači na obrazovce kolegy. Musíte si vytvořit potřebné účty. Použijte zabezpečené připojení!
Nastavte si SSH klíče tak, abyste se na kolegův počítač mohli připojit bez zadávání hesla.
Pomocí příkazu scp vytvořte kopii svého domovského adresáře v adresáři /var/tmp na kolegově počítači. Před zahájením přenosu data archivujte a zkomprimujte! Připojte se na vzdálený systém příkazem ssh, rozbalte zálohu a pomocí sftp jeden ze souborů zkopíruj-te zpět na svůj počítač.

Bezpečnost

Sestavte seznam otevřených (poslouchajících) portů na svém počítači.
Řekněme, že budete chtít provozovat webový server. Které služby vypnete? Jak to uděláte?
Nainstalujte dostupné aktualizace.
Jak můžete zjistit, kdo je k vašemu systému připojen?
Vytvořte něco, co vám pravidelně každý měsíc připomene, že máte změnit své heslo i heslo uživatele *root*.

Zvuk a video

V této kapitole budeme (stručně, protože oblast zvuku a videa je velmi rozsáhlá) hovořit o následujících tématech:

Konfigurace zvukové karty.
Přehrávání CD, kopírování CD.
Přehrávání zvukových souborů.
Ovládání hlasitosti.
Video a televize.
Záznam zvuku.

Základy přehrávání zvuku

Instalace

S největší pravděpodobností už máte systém nainstalován i s ovladači zvukové karty a potřebná konfigurace proběhla při instalaci. Podobně kdybyste někdy zvukovou kartu měnili, většina systé-mů obsahuje nástroje, které umožňují snadnou instalaci a konfiguraci těchto zařízení. Většina moderních karet s architekturou plug-and-play by měla být detekována automaticky. Pokud při konfiguraci uslyšíte přehrávaný ukázkový zvuk, klepněte prostě na OK a všechno se správně nastaví.

Pokud nedojde k automatické detekci karty, konfigurátor vám pravděpodobně nabídne seznam zvukových karet a doplňků, ze kterého si budete moci vybrat. Poté budete muset nejspíš zvolit správný vstupně-výstupní port, přerušení a kanál DMA – týká se to hlavně starých ISA karet. Pří-slušné informace byste měli najít v dokumentaci ke zvukové kartě. Pokud používáte dualboot systém s MS Windows, naleznete potřebné údaje pravděpodobně i v Ovládacích panelech Win-dows.

Nezdaří-li se automatická detekce

Pokud máte zvukovou kartu, která není standardně podporována, budete muset použít slo
žitější postupy. Tyto postupy najdete v dokumentu Linux Sound HOWTO,
http://www.tldp.org/HOWTO/Sound-HOWTO/index.html.

Ovladače a architektura

V současné době existují dvě architektury zvukového systému: starší Open Sound System, OSS, který pracuje na všech systémech unixového typu, a novější Advanced Linux Sound Architekture, ALSA, která je primárně určena pro Linux. ALSA nabízí větší množství funkcí a usnadňuje vývoj ovladačů. My se zaměříme právě na systém ALSA.

V současné době je podporována drtivá většina nejpoužívanějších zvukových chipsetů. Nepod-porovány zůstávají profesionální high-end řešení a dále některé karty vyvíjené výrobci, kteří odmí-tají zveřejnit dokumentaci ke svým chipsetům. Seznam podporovaných zařízení naleznete na stránkách projektu ALSA na adrese *http://www.alsa-project.org/*.

Konfigurace systémů, na nichž je nainstalována ALSA, se provádí příkazem alsaconf. Navíc jed-notlivé distribuce obvykle nabízejí vlastní nástroje pro nastavení zvukových karet, tyto nástroje dokonce mohou integrovat starší i novější způsoby práce se zvukovými zařízeními (v závislosti na detekované zvukové kartě).

Přehrávání zvuku a videa

Přehrávání a kopírování CD

Většina distribucí obsahuje balík *cdp* a nabízí program cdp nebo cdplay, textový přehrávač CD. Grafická pracovní prostředí obvykle obsahují i vlastní grafické nástroje, například gnome-cd v prostředí Gnome, které lze spustit přímo z nabídky prostředí, nebo kscd pro KDE.

Nezapomínejte na rozdíly mezi zvukovými CD a datovými CD. Zvuková CD můžete poslouchat i bez toho, že byste je připojovali do souborového systému. Je to dáno tím, že data na takovém CD nejsou uložena v podobě linuxového souborového systému, prostřednictvím přehrávače CD se tato data načítají a přímo předávají na zvukový výstup. Pokud ovšem máte na CD uloženu hudbu v podobě souborů .mp3, budete muset takové CD nejprve připojit a následně je pak pře-hrát některým z programů, o nichž

budeme hovořit dále. Připojení CD k souborovému systému je popsáno v kapitole „Instalace dodatečných balíčků z instalačních CD“.

Nástroj cdparanoia ze stejnojmenného balíku čte přímo z CD zvuková data a bez analogové kon-verze je zapisuje do souboru nebo roury v jiném formátu, nejčastěji ve formátu .wav. Většina distri-bucí pak obsahuje různé nástroje pro převod do dalších formátů, jako je například .mp3, celou řadu podobných programů je také možno si stáhnout. Projekt GNU nabízí několik nástrojů pro přehrávání, ripování a překódování CD i databázové manažery pro správu obsahu médií. Podrob-nosti naleznete ve Free Software Directory sekce Audio na adrese *http://directory.fsf.org/audio/*.

Ripování zvukových CD je usnadněno díky nástrojům, jako je kaudiocreator z balíku KDE. Samo-zřejmostí je integrovaná nápověda; program naleznete na všech systémech, kde je nainstalován balíček kdemultimedia. Populární je také program Grip pro prostředí Gnome. Dobře zpracované informace o tomto tématu najdete například na stránce *http://www.linuxexpres.cz/praxe/hudebni-cd-na-minimum-1-cast.*

Vypalování CD je popsáno v kapitole „Vytvoření kopie na CD vypalovačce“.

Přehrávání zvukových souborů Soubory mp3

Oblíbený formát .mp3 je v Linuxu široce podporován. Většina distribucí obsahuje hned několik programů, které dokážou tyto soubory přehrát. Velmi oblíbenou aplikací je mimo jiné přehrávač XMMS, zobrazený na následujícím obrázku, jehož obliba pramení mimo jiné i z toho, že vypadá a chová se velmi podobně jako odpovídající nástroje ve Windows, konkrétně program Winamp. Nástupcem XMMS je přehrávač Audacity.

Přehrávač XMMS

Dalšími oblíbenými přehrávači jsou AmaroK, KDE aplikace, jejíž obliba se stále zvyšuje, a Mplay-er, který dokáže přehrávat i filmy. Na výsluní přízně uživatelů se poslední dobou derou právě komplexní přehrávače jako Amarok či Kaffeine.

> Omezení
>
> Na některých distribucích není možné po standardní instalaci soubory MP3 přehrávat, je to dáno licenčními omezeními příslušných nástrojů. V těchto případech si budete muset pře-hrávače MP3 dodatečně nainstalovat. Viz například HOWTo pro Fedoru v této knize.

Další formáty

Následující (neúplný) seznam představuje některé další oblíbené zvukové formáty a programy pro práci s nimi:

Ogg Vorbis – svobodný zvukový formát, příslušné nástroje naleznete v adresáři Free Soft-ware, *http://directory.fsf.org/audio/ogg/*. Některé z nich mohou být součástí i vaší distribu-ce. Formát vznikl v důsledku patentových ochran formátu MP3.
Real audio a video – přehrávač realplay od RealNetworks, *http://www.real.com/*.
SoX neboli Sound eXchange – převaděč zvukových formátů, obsahuje i přehrávač play. Dokáže přehrávat formáty .wav, .ogg a celou řadu dalších včetně přímých binárních for-mátů.
AlsaPlayer z projektu Advanced Linux Sound Architecture, viz *http://www.alsaplayer.org/*.
mplayer – přehrává téměř cokoliv včetně souborů MP3. Více viz *http://mplayerhq.hu.* GUI nástavby se jmenují GMPlayer a KMPlayer.
hxplay – přehrává RealAudio a RealVideo, mp3, mp4, Flash, wav a celou řadu dalších, viz *https://helixcommunity.org/*. Ne všechny komponenty tohoto přehrávače jsou zcela svo-bodné.
rhytmbox – přehrávač založený na platformě GStreamer, přehrává veškeré formáty, které jsou podporované touto platformou (což jsou údajně všechny formáty). Viz *http://www.gnome.org/projects/rhythmbox/* a *http://gstreamer.freedesktop.org/*.

Podrobnosti o jednotlivých nástrojích a jejich použití naleznete v dokumentaci ke svému systému a na manuálových stránkách.

> Pokud tyto aplikace v systému nemáte
>
> Řada výše popsaných nástrojů a aplikací jsou doplňkové programy. Je možné, že ve stan-dardní instalaci systému nebudou nainstalovány, mohou však být k dispozici jako volitel-né balíky distribuce. V některých případech nemusí být jednotlivé programy vůbec sou-částí vaší distribuce, v takových případech si je budete muset stáhnout přímo z příslušných stránek. Například vynikajícím zdrojem balíčků s různými v distribucích nezahrnutými pro-gramy jsou pro systémy RedHat/CentOS/Fedora stránky *http://dag.wieers.com/home-made/apt/*, pro Mandriva Linux existují stránky *http://easyurpmi.zarb.org/* a podobně.

Ovládání hlasitosti

Programy aumix a alsamixer jsou běžné textové nástroje pro ovládání hlasitosti. Nastavené hod-noty se mění pomocí kurzorových

kláves. Program alsamixer má i grafické rozhraní, které je možné spustit z nabídky Gnome nebo přímo příkazem gnome-alsamixer. V prostředí KDE plní stejnou funkci nástroj kmix.

Bez ohledu na to, jak hudbu či cokoliv jiného přehráváte, nezapomínejte, že kolem mohou být i lidé, kteří na vás či váš počítač nejsou zvědaví. Zejména v kancelářském prostředí proto buďte ohleduplní. A samozřejmě používejte kvalitní sluchátka.

Záznam

Pro záznam zvuku a videa je k dispozici mnoho různých nástrojů. Pro záznam zvuku z příkazo-vého řádku můžete použít například příkaz arecord:

alexey@russia:~> arecord /var/tmp/myvoice.wav Recording WAVE '/var/tmp/myvoice.wav' : Unsigned 8 bit, Rate 8000 Hz, Mono Aborted by signal Interrups...

Záznam přerušíte stiskem kláves Ctrl+C. Zaznamenaný vzorek můžete přehrát příkazem play. Tento postup můžete použít jako vhodný test před tím, než se pokusíte nastavovat aplikace, které vyžadují zvukový vstup – například Voice over IP (VoIP). Nezapomeňte, že je nutné mít aktivo-vaný mikrofonní vstup. Pokud se sami neslyšíte, zkontrolujte zvuková nastavení. Velmi často se stává, že je mikrofonní vstup potlačen nebo ztlumen na minimální úroveň. Nastavení je možné snadno změnit programem alsamixer, případně distribučním grafickým rozhraním zvukového subsystému.

Přehrávání videa, streamů a sledování televize

K dispozici jsou různé přehrávače:

- xine: svobodný videopřehrávač (používá ho i oblíbený Kaffeine).
- ogle: přehrávač DVD.
- okle: KDE verze přehrávače ogle.

mplayer: linuxový přehrávač videa (s nástavbami GMPlayer nebo KMplayer).
gstreamer: multimediální platforma založená na knihovně GStreamer, obsahující souvise-jící nástroje pro záznam zvuku a videa, jejich editaci a přehrávání. Jde o součást Gnome. Více viz *http://www.gstreamer.net/*.
totem: přehrává zvuk i video, CD, VCD a DVD.
realplay: od RealNetworks.
hxplay: alternativa k programům Real, viz *https://helixcommunity.org/*.

Je velmi pravděpodobné, že některé z uvedených nástrojů naleznete přímo v grafické nabídce systému.V rámci LDP (*http://www.tldp.org/*) naleznete dokument DVD Playback HOWTO, viz stránky*http://www.tldp.org/HOWTO/DVD-Playback-HOWTO/index.html,* který popisuje různé nástrojepro přehrávání DVD.

Chcete-li sledovat televizi, jsou k dispozici například následující nástroje – a mimo ně i celá řada dalších, které umožňují přehrávat a zachytávat TV, video a další streamy:

tvtime – vynikající program umožňující správu stanic, spolupráci s teletextem, filmový režim a mnoho dalších, viz *http://tvtime.sourceforge.net/*.
zapping – televizní přehrávač pro Gnome.
xawtv – televizní přehrávač pro X Window.

Internetová telefonie

Co to je?

Internetová telefonie, častěji též Voice over IP (VoIP) či digitální telefonie, umožňuje účastníkům přenos hlasových toků přes síť. Největší výhodou je, že data jsou přenášena po univerzální síti – Internetu, na rozdíl od klasické telefonie, kdy se pro přenos hlasu používají vyhrazené přenoso-vé linky. Tyto dvě sítě spolu lze za jistých okolností propojit, v současné době však takové řeše-ní není nijak standardizováno. Jinak řečeno – je velmi pravděpodobné, že pomocí internetové tele-fonie nebudete moci komunikovat s účastníky klasické telefonní služby.

V současné době je sice zdarma k dispozici celá řada aplikací, jak otevřených, tak proprietárních, nicméně internetová telefonie má stále své zásadní nevýhody. Nejdůležitější je, že systém je nespo-lehlivý, může být pomalý a spojení může být značně rušeno. Nelze ji proto chápat jako náhradu klasické telefonie – pomyslete například na tísňová volání. Někteří poskytovatelé se snaží situaci různými opatřeními řešit, nemáte však nikdy záruku, že se vám podaří spojit se s protistranou.

Co budete potřebovat? Strana serveru

V první řadě budete potřebovat poskytovatele, který takovou službu nabízí. V rámci takovýchto služeb může být poskytováno i

propojení s klasickou telefonní sítí a služby nemusí být zdarma. Tyto takzvané „úplné telefonní služby“ nabízí celá řada poskytovatelů, mnoho informací ohledně situace v České republice naleznete například na adrese *http://www.telefonujeme.com/*.
Pokud byste chtěli provozovat vlastní server hlasových služeb, mohou vás zajímat programy Aste-risk (*http://www.asterisk.org/*), Gizmo (*http://gizmoproject.com/*), případně GnomeMeeting (*http://www.gnomemeeting.org/*), abychom se zmínili alespoň o některých.

Strana klienta

Výběr vhodné klientské aplikace závisí na konfiguraci sítě. Pokud máte přímé připojení k Inter-netu, neměl by být s připojením problém za předpokladu, že víte, ke kterému serveru se chcete připojit, a máte případně k dispozici potřebné jméno a heslo.

Pokud jste ovšem připojeni přes firewall se službou překladu adres (NAT), nemusí některé pro-gramy fungovat, protože vnější svět vidí pouze IP adresu vašeho firewallu, a nikoliv přímo adre-su vašeho počítače, která vůbec nemusí být z Internetu dostupná. IP adresy začínající na 10.,
192.168. a některé další jsou neveřejné, a tedy nedostupné. Použitelnost služby pak závisí na kon
krétním protokolu.
Dalším omezujícím faktorem může být přenosová kapacita – některé aplikace jsou optimalizová
ny tak, aby měly minimální přenosové nároky, jiné mohou mít nároky vysoké. To závisí na pou
žívaném kodeku.

Mezi nejrozšířenější aplikace patří klient služby Skype, který nabízí rozhraní podobné jiným komu-nikačním službám, a X-Lite, volně dostupná verze softwarového telefonu XTen, který vypadá jako mobilní telefon. I když jsou tyto programy k dispozici zdarma a jsou velmi oblíbené, nejsou ote-vřené – používají proprietární protokoly a jsou k dispozici pouze jako binární balíčky, nikoliv ve formě zdrojových kódů.

Zdarma a zároveň otevřený VoIP klient je například GnomeMeeting/Ekiga (*http://www. gnomemeeting.org/*) nebo KPhone (*http://sourceforge.net/projects/kphone*).

Klientský hardware

I když váš počítač (zejména v případě laptopu) může být vybaven vestavěným mikrofo
nem, mnohem lepších výsledků dosáhnete s externí náhlavní sadou. Máte-li na výběr, zvol
te USB provedení, které funguje nezávisle na ostatním audio hardwaru.

Trh s VoIP aplikacemi se stále rozrůstá. Aktuální stav se pokouší dokumentovat projekt *http://www.voip-info.org/wiki/*.

Shrnutí

Platforma GNU/Linux nabízí plnou podporu multimédií. Podporována je široká množina zvukových karet, televizních karet, náhlavních souprav, mikrofonů, CD a DVD mechanik. Seznam aplikací je prakticky nekonečný.

Cvičení

Dokážete na svém systému přehrát CD?
Pomocí příkazu locate zkuste v systému najít některý ze souborů .wav, .ogg nebo .mp3. Přehrajte jej.
Dokážete regulovat hlasitost přehrávání?
Dokážete provést záznam zvuku? (K tomuto cvičení potřebujete mikrofon.) Dokážete záznam přehrát?
V grafickém rozhraní ověřte, jaké aplikace máte k dispozici pro přehrávání zvuku, záznam a regulaci hlasitosti.

Následující cvičení vám umožní lépe se seznámit se svým systémem.

Z nabídky Gnome či KDE otevřete panel pro nastavení zvuku. Zkontrolujte, že máte k systému připojeny reproduktory či sluchátka, a nastavte si vhodnou hlasitost. Samozřejmě použijte správný ovládací panel podle toho, zda váš systém je či není založen na architektuře ALSA.
Máte-li mikrofon, zkuste zaznamenat vlastní hlas. Zkontrolujte, zda není úroveň záznamu příliš vysoká, pak by byl záznam zkreslený, případně by zbytečně obsahoval i šum v poza-dí. V příkazovém řádku můžete pro záznam a přehrání vyzkoušet příkazy arecord a aplay.
Zkuste v systému nalézt nějaké zvukové soubory a přehrát je.
Vložte do mechaniky zvukové CD a přehrajte je.
Společně s kolegou si vyzkoušejte komunikaci pomocí VoIP.
Můžete poslouchat internetové rádio?
Máte-li DVD přehrávač a nějaký film na DVD, zkuste jej přehrát.

Kam dál

V této příloze uvádíme přehled užitečných knih a dalších pramenů.

Užitečné knihy

Linux obecně (v češtině)

Barrett, D. J.: Linux – kapesní přehled, Computer Press 2006.
Barrett, D. J.; Silverman, R. E.: SSH kompletní průvodce, Computer Press 2005.
Dobšíček, M.; Ballner, R.: Linux – bezpečnost a exploity, Kopp 2004.
Herborth, Chris: Unix a Linux, názorný průvodce, Computer Press 2006.
Kysela, M. a kolektiv: 333 tipů a triků pro Linux, Computer Press 2005.
Kysela, M.: Přecházíme na Linux, Computer Press 2003.
Nemeth, Evi; Synder, Garth; Hein Trent R.: Linux Kompletní příručka administrátora, Computer Press 2004.
Sobell, Mark G.: Mistrovství v Linuxu Příkazový řádek, shell, programování, Computer Press 2007.
Stanfield, Vicki; Smith, Roderick W.: Správa operačního systému Linux, Softpress 2002.
Toxen, Bob: Bezpečnost v Linuxu, Computer Press 2003.
Vychodil, Vilém: Linux – příručka českého uživatele, Computer Press 2003.

Užitečné webové stránky

Obecné informace

The Linux Documentation Project, *http://www.tldp.org/* – dokumentace, manuály, HOWTO, FAQ.
LinuxQuestions.org, *http://www.linuxquestions.org/* – fórum, download, dokumentace a mnoho dalších.
Google for Linux, *http://www.google.com/linux* – specializovaný vyhledávač.
Deja, *http://groups.google.com/* – archiv diskusních skupin včetně hierarchie comp.os.linux (totéž umí pro české zdroje *http://usenet.jyxo.cz*).
Linux HQ, *http://www.linuxhq.com/* – databáze zdrojových kódů, patchů a dokumentace různých verzí linuxových jader.

Informace ke konkrétním architekturám

Linux PPC, *http://www.linuxppc.org* – Linux na architektuře Power PC (např. Apple Powe-rPC, PowerMac, Amiga, IBM ThinkPad/PowerSeries/RS/6000, Motorola, …).
AlphaLinux, *http://www.alphalinux.org/* – Linux na architektuře Alpha (např. Digital Work-station).
Linux-MIPS, *http://www.linux-mips.org/wiki/Main_Page* – Linux na architektuře MIPS (např. SGI Indy).
MkLinux, *http://www.mklinux.org* – Linux na Apple.

Distribuce

The Fedora Project, *http://fedora.redhat.com/*.
Mandriva, *http://www.mandriva.com/*.
Debian, *http://www.debian.org/*.
Slackware, *http://www.slackware.com/*.▪openSUSE, *http://www.opensuse.org/*.
LinuxISO.org, *http://www.linuxiso.org/* – ISO obrazy CD různých distribucí.
Knoppix, *http://www.knoppix.org/* – distribuce spouštěná z CD, nemusíte nic instalovat.
DistroWatch.com, *http://distrowatch.com/* – vyberte si distribuci podle svého vkusu.

Software

Freshmeat, *http://freshmeat.net/* – nové programy, archivy programů.
OpenSSH, *http://www.openssh.org/* – projekt Secure Shell.

OpenOffice.org, *http://www.openoffice.org/* – kancelářský balík kompatibilní s MS Office.
KDE, *http://www.kde.org* – grafické prostředí.
GNU, *http://www.gnu.org* – GNU programy.
Gnome, *http://www.gnome.org* – grafické prostředí.
Ximian, *http://www.ximian.com* – Ximina Gnome, balíčkovací systém Red Carpet, Opera, CodeWeavers, Loki Demos, Evolution, systémové balíčky a další.
RPM find, *http://www.rpmfind.net* – všechny RPM balíčky.
Samba, *http://www.samba.org* – sdílení souborů a tiskáren s MS Windows.
OpenLDAP, *http://www.openldap.org* – server, klient a nástroje projektu OpenLDAP, FAQ a další dokumentace.
Sendmail, *http://www.sendmail.org* – podrobné informace o funkcích systému Sendmail včetně příkladů konfigurace.
Netfilter, *http://www.netfilter.org/* – informace o iptables.
GIMP, *http://www.gimp.org/* – informace o grafickém editoru GIMP.
SourceForge, *http://sourceforge.net* – mnoho softwarových projektů.

Užitečné stránky v češtině

http://proc.linux.cz – přehledná stránka o základních vlastnostech Linuxu, odkazy a množ-ství dalších zajímavých informací.
http://www.debian.cz – webové stránky distribuce Debian.
http://www.fedora.cz – české stránky distribuce Fedora Core.
http://www.mandrivalinux.cz – české stránky distribuce Mandriva Linux.
http://www.suseportal.cz – stránky distribuce openSUSE.
http://www.ubuntu.cz – stránky distribuce Ubuntu.

Portály o systému Linux: *http://www.root.cz, http://www.abclinuxu.cz/, http://www.linuxsoft.cz.*
Časopis LinuxEXPRES: *http://www.linuxexpres.cz.*

Srovnání příkazů DOSu a Linuxu

V této příloze srovnáváme příkazy DOSu a jejich linuxové ekvivalenty. Zejména pro usnadnění orientace začínajícím uživatelům Linuxu, kteří mají znalosti systému Win-dows, shrnuje následující tabulka základní příkazy MS-DOS a jejich linuxové protějšky. Většina pří-kazů v Linuxu používá celou řadu voleb. Více informací se dozvíte na informačních nebo manu-álových stránkách příslušného příkazu.

Příkaz DOSu	Příkaz Linuxu
	nebo
	(nebo jiný editor)
	nebo
	nebo

Funkce shellu

Tato příloha shrnuje známé běžné funkce shellu (společné v různých shellech) a také rozdíly mezi některými shelly.

Společné funkce

Následující funkce jsou standardní ve všech shellech. Příkazy stop, suspend, jobs, bg a fg jsou dostupné pouze na systémech, které podporují řízení úloh.

Příkaz	Popis
>	Přesměrování výstupu
>>	Přidání do souboru
<	Přesměrování vstupu
<<	Přesměrování vstupu
|	Přesměrování do roury
&	Spustí program na pozadí
;	Odděluje více příkazů na řádku
*	Zastupuje libovolné znaky v názvu souboru
?	Zastupuje jeden znak
[ ]	Zastupuje vyjmenované znaky
( )	Spustí se v subshellu
``	Bude nahrazeno výstupem uvedeného příkazu
" "	Částečné uzavření (umožňuje expanzi proměnných a příkazů)
'	Úplné uzavření (bez expanze)
\	ESCapuje následující znak
$var	Použije hodnotu proměnné
$$	ID procesu
$0	Název příkazu
$n	n-tý parametr (pro n 0 až 9)
$*	Všechny parametry jako jeden řetězec
#	Začátek komentáře
	Provedení na pozadí
	Ukončuje smyčky

Příkaz	Popis
	Změna adresáře
	Pokračování další iterací smyčky
	Zobrazuje výstup
	Vyhodnocuje parametry

Spustí nový shell
Provedení na popředí
Ukáže aktivní úlohy
Ukončuje běžící úlohy
Změní skupinu
Posouvá parametry
Pozastaví úlohu na pozadí
Pozastaví úlohu na popředí
Změří dobu běhu příkazu
Určuje práva nových souborů
Ruší proměnnou nebo definici funkce
Čeká na skončení úlohy na pozadí

Společné funkce shellů

Odlišné funkce

Následující tabulka shrnuje základní rozdíly mezi standardním shellem (sh), Bourne Again shel-lem (bash), Korn shellem (ksh) a C shellem (csh).

Kompatibilita shellů

Bourne Again SHell je nadmnožinou sh, takže všechny příkazy použitelné v sh fungují také v bashi – ne však opačně; bash má mnoho dalších vlastních funkcí a funkcí přejatých z jiných shellů.

Turbo C shell je nadmnožina csh, všechny příkazy csh fungují v tcsh, neplatí to však opačně.

sh bash ksh csh Význam/Popis

$	$	$	%	Standardní prompt
				Nucené přesměrování
soubor	soubor	soubor	soubor	Přesměrování stdout
	nebo soubor			a stderr do souboru
				Expanduje prvky ze seznamu
``	``nebo		``	Nahradí se výstupem
$HOME	$HOME	$HOME	$home	Domovský adresář
	~	~	~	Symbol domovského adresáře
	~+, ~-,	~+, ~-	=-, =N	Přístup k zásobníku adresářů

sh	bash	ksh	csh	Význam/Popis
				Přiřazení hodnoty proměnné
			hodnota	Nastavení proměnné prostředí
	${nnnn}	${nn}		Odkazování se na více než 9 parametrů
"$@"	"$@"	"$@"		Všechny parametry jako samostatná slova
$#	$#	$#	$#argv	Počet parametrů
$?	$?	$?	$status	Návratový kód posledního spuštěného

				programu
$!	$!	$!		PID procesu naposledy spuštěného na pozadí
$-	$-	$-		Aktuální volby
soubor	soubor nebo		soubor	Čte příkazy ze souboru
	"			Příkazu se přiřadí alias
			nebo	Volba mezi alternativami
				Konec smyčky
				Konec nebo
				Ukončení s návratovým kódem
/	/	/		Smyčka přes proměnné
	,			Ignoruje substituční znaky při generování názvů souborů
				Zobrazuje hashované příkazy
				Zapamatuje se umístění příkazů
				Zapomene umístění příkazů
				Výpis historie příkazů
	+nebo			Znovuprovedení předchozího příkazu
				Znovuprovedení posledního příkazu začínajícího na „str“ Nahradí text „x“ textem „y“ v posledním příkazu začínají-cím na „cmd“ a příkaz prove-de
				Jednoduchý test
				Konec příkazu
				Nastavení limitu prostředků

sh	bash	ksh	csh	Význam/Popis
				Výpis aktuálního adresáře Čtení z terminálu
				Ignorují se přerušení Zrušení aliasů
/	/	/		Začátek smyčky Začátek smyyčky

Odlišné funkce různých shellů

Bourne Again SHell má ještě celou řadu dalších funkcí, které v tabulce nejsou uvedeny. Tabulka slouží pouze jako ukázka toho, jak tento shell integruje užitečné myšlenky z jiných shellů – ve sloupci bash není žádné pole prázdné. Informace o funkcích, které umí jenom bash, naleznete na

informačních stránkách v části „Bash Features“.

Další informace: Minimálně byste si měli přečíst manuál ke svému shellu. Doporučujeme vám pří-kaz info bash.

ČÁST II

Příručka správce operačního systému

Úvod

Na počátku byl soubor nesličný a pustý, a prázdno se vznášelo nad povrchem bitů. A Ruka Auto-rova dosedla na povrch klávesnice – i řekl Autor: „Buďtež slova!“ A byla slova.

Příručka správce operačního systému Linux popisuje ty aspekty používání operačního systému, jež se vztahují k jeho správě. Je určena lidem, kteří o správě operačního systému neví zhola nic (ptají se teď, co to je), avšak zvládají přinejmenším základy jeho běžného užívání. Nenaleznete zde návod, jak Linux instalovat. Instalace systému je podrobně popsaná v dokumentu „Průvodce insta-lací a začátky“. Další informace o dokumentaci k systému Linux jsou uvedeny níže.

Administrací (správou) systému rozumíme všechny činnosti, které je nutno pravidelně vykonávat, aby počítačový systém zůstal v provozuschopném stavu. Zahrnuje například zálohování souborů (a v případě potřeby jejich obnovování), instalaci nových programů, vytváření uživatelských účtů (a jejich mazání v případě, že jsou nepotřebné), kontroly a opravy případných poškození systému souborů a další. Když si počítač představíte jako dům, pak by správou systému byla jeho údržba. Ta by zahrnovala například úklid, zasklívání rozbitých oken a další podobné věci.

Příručka je strukturována tak, že většinu kapitol můžete číst nezávisle na sobě. Když například hledáte nějaké informace o zálohování, stačí, když si přečtete příslušnou kapitolu. Přesto manuál zůstává především učebnicí a jakýmsi průvodcem a můžete jej číst od začátku do konce.

Nelze předpokládat, že by tato knížka pokryla celou problematiku administrace systému. Správce systému bude potřebovat řadu další dokumentace operačního systému Linux. Koneckonců, admi-nistrátor je v podstatě jenom uživatel, který má zvláštní práva a povinnosti. Velmi významným pra-menem jsou manuálové stránky, po kterých by správce měl sáhnout pokaždé, když si není funk-cí některého příkazu zcela jist. Nevíte-li, jaký příkaz použít, vyzkoušejte příkaz apropos. Další podrobnosti viz manuálová stránka tohoto příkazu.

Tato příručka je zaměřena především na operační systém Linux, ale v obecných principech může být užitečná i pro správce jiných unixových systémů. Bohužel je mezi různými verzemi Unixu tolik rozdílů (a o správě systému to platí dvojnásob), že není prakticky možné postihnout všechny známé varianty. Je totiž obtížné – vezmeme-li v potaz způsob, jakým se Linux vyvíjí – pokrýt i všechny možnosti jen tohoto operačního systému.

Neexistuje jediná oficiální distribuce Linuxu od jediného výrobce. Různí lidé používají různá nastavení a konfigurace. Navíc si řada uživatelů vytváří své vlastní. Proto tato kniha není zaměřená na některou z konkrétních distribucí. V rámci možností se v příručce snažíme upozornit na některé odlišnosti a objasnit i jiné možné alternativy. Seznam distribucí a některé rozdíly mezi nimi viz

http://en.wikipedia.org/wiki/Comparison_of_Linux_distributions.

Než abychom podali strohý seznam „pěti jednoduchých kroků“ pro řešení každého úkolu, dáváme přednost popisu základních principů, tedy objasnění toho, jak věci doopravdy fungují. V knize proto najdete hodně informací, které nejsou nezbytné pro každého. Takovéto části jsou v textu označeny a v případě, že používáte systém s předem nastavenou konfigurací, můžete je klidně přeskočit. Pochopitelně, přečtete-li si knihu celou, proniknete do systému hlouběji, a pak by pro
vás mohly být o něco příjemnější i jeho používání a jeho správa. Porozumění představuje v případě Linuxu základní podmínku úspěchu. Tato kniha by mohla být pouhým seznamem návodů – jenže co byste dělali tváří v tvář problému, na nějž byste zde návod nenalezli. Pokud vám ale nabídneme vysvětlení principů, nejsou návody nutné – vyplynou samy ze znalosti věci.

Tak jako vše ostatní spojené s vývojem Linuxu, byla i tato práce založena na principu dobrovol-nosti. Pustili jsme se do ní, protože jsme si mysleli, že by to mohla být zábava. Dalším důvodem byl pocit, že je potřeba tuto práci udělat. Přesto – jako konečně u každé dobrovolné práce – jsou určité hranice nasazení a úsilí, které můžete vynaložit. Navíc vás omezuje také to, kolik vědomostí a zkušeností máte. Přirozeně, manuál není tak dobrý, jak by mohl být v případě, že by přišel někdo s kouzelnou hůlkou a dobře zaplatil za jeho napsání. Pak by bylo možné strávit i několik dalších let jeho zdokonalováním. Samozřejmě si myslíme, že je celkem povedený, nicméně berte to jako varování.

Je jeden konkrétní bod, ve kterém jsme manuál dost „ořezali“ – není v něm vyčerpávajícím způ-sobem popsaná řada věcí, které již jsou podrobně zdokumentované v jiných volně dostupných příručkách. Vztahuje se to zvlášť na dokumentaci k jednotlivým programům. Neuvádíme například všechny podrobnosti použití programu mkfs. Popisujeme jenom funkci programu a pouze tolik z jeho dalších možností, kolik je potřeba pro dosažení účelu této knihy. Laskavého čtenáře, jenž hledá podrobnější informace, odkazujeme na onu další dokumentaci. Převážná většina dokumen-tů, na které se odvoláváme v odkazech, je součástí úplné sady dokumentace k operačnímu syste-mu Linux.

O této části

„Pouze dvě věci jsou nekonečné, vesmír a lidská hloupost. Tou první si nejsem tak docela jist.“ Albert Einstein.

Poděkování Joanna děkuje

Při práci na dokumentu nám přímo či nepřímo pomáhalo mnoho lidí. Rádi bychom zvlášť podě-kovali Mattu Welshovi za inspiraci a vedení projektu LDP; Andy Oramovi za to, že nás znovu a znovu zaměstnával řadou velmi podnětných připomínek; Olafu Kirschovi za to, že nám doká-zal, že vše lze zvládnout; Adamu Richterovi z Yggdrasil a dalším za to, že nám ukázali, že tato práce může být zajímavá i pro jiné lidi.

Stephen Tweedie, H. Peter Anvin, Rémy Card a Theodore Ts’o odvedli kus práce, kterou jsme si formou odkazů a referencí „zapůjčili“ (tím pádem je naše kniha na pohled tenčí a o to víc půso-bivá): Sem patří porovnání souborových systémů xia a ext2, seznam zařízení nebo popis soubo-rového systému ext2. Tyto kapitoly jsme z knihy vyřadili. Za toto jsme vděční vůbec nejvíc a záro-veň se velmi omlouváme za předchozí verze manuálu, které občas v některých oblastech postrá-daly odpovídající úroveň.

Kromě toho patří náš dík Marku Komarinskému za jeho materiály z roku 1993 i mnoho dalších sloupků v Linux Journalu, jež se týkaly problematiky správy systému. Jsou velmi informativní a inspirující.

Dostali jsme množství užitečných připomínek od velkého počtu dalších lidí. Díky malé černé díře v našem archivu nelze dohledat všechna jména, takže alespoň některá z nich (v abecedním pořadí): Paul Caprioli, Ales Cepek, Marie-France Declerfayt, Dave Dobson, Olaf Flebbe, Helmut Geyer, Larry Greenfield a jeho otec, Stephen Harris, Jyrki Havia, Jim Haynes, York Lam, Timothy Andrew Lister, Jim Lynch, Michael J. Micek, Jacob Navia, Dan Poirier, Daniel Quinlan, Jouni K. Seppänen, Philippe Steindl, G. B. Stotte. Omlouváme se všem, na které jsme zapomněli.

Stephen děkuje

Chtěl bych poděkovat Larsovi a Joanně za jejich vyčerpávající práci na příručce. V příručce, jako je tato, se téměř vždy najdou přinejmenším drobné nepřesnosti. Kromě toho se v ní objeví kapitoly, které postupně zastarávají. Pokud cokoliv z toho postřehnete, pošlete mi las
kavě e-mail na adresu <bagpuss@debian.org.NOSPAM>. Akceptuji připomínky v jakémkoliv for-mátu – diff, text, HTML, cokoliv. Nikomu nechci bránit pomoci mi s prací nad tímto textem. Mnohokrát děkuji Helen Topping Shawové za práci s červenou tužkou, díky níž je příručka dale-
ko lepší, než by byla jinak.

Alex děkuje

Rád bych poděkoval Larsovi, Joanně a Stephenovi za skvělou práci, kterou za ty roky odvedli na tomto dokumentu. Pouze doufám, že můj příspěvek bude plnohodnotným pokračováním práce, kterou začali.

Na cestě světem „Windows-Free“ mně pomáhalo mnoho lidí. Cítím ovšem, že osoba, které musím poděkovat nejvíc, je můj první skutečný rádce ve věci UN*X, Mike Velasco. Ještě než se SCO stalo „sprostým slovem“, Mike mně pomohl zvládnout manuálové stránky tar, cpio a mnoho dalších. Díky, Miku! Jsi nejlepší.

Přehled operačního systému Linux

A viděl Bůh vše, což učinil, a aj, bylo velmi dobré. -- Bible kralická, 1613, Genesis 1, 31

Tato kapitola podává zevrubný přehled o operačním systému Linux. V první části jsou popsány nejdůležitější ze služeb, jež systém nabízí. Další části se bez přílišných podrobností zabývají pro-gramy, které popsané služby realizují. Cílem kapitoly je podat výklad principů systému jako celku s tím, že každé téma bude podrobněji probráno později, na jiném místě knihy.

Různé části operačního systému

Operační systém typu Unix se skládá z *jádra* a *systémových programů*. Kromě toho pro různou běžnou práci existují *aplikační programy*. Jádro je srdcem operačního systému. Často bývá jádro považováno za celý operační systém, to ale není pravda. Operační systém nabízí mnohem více služeb než jen čisté jádro.

Udržuje záznamy o souborech na disku, spouští programy, řídí jejich současný běh, přiděluje paměť a další technické prostředky různým procesům, přijímá a odesílá pakety z a do počítačové sítě a tak dál. Jádro systému samotné toho dělá velmi málo, ale poskytuje základní služby různým nástrojům, pomocí kterých mohou být realizovány všechny ostatní služby. Jádro rovněž hlídá, aby nikdo nemohl přistupovat k hardwarovým zařízením přímo. Když chtějí uživatelé a procesy pou-žívat technické prostředky, musí používat nástroje, které nabízí jádro systému. Tímto způsobem je zabezpečená i vzájemná ochrana uživatelů. Nástroje jádra systému, o nichž byla řeč, lze využívat prostřednictvím *systémových volání*. Podrobnější informace o systémových voláních uvádí sekce 2 manuálových stránek.

Systémové programy realizují služby, které se vyžadují od operačního systému. Využívají přitom nástroje, které nabízí jádro systému. Systémové i všechny ostatní programy běží jakoby „na povr-chu“ jádra. Říká se tomu *uživatelský režim*. Rozdíl mezi systémovými a aplikačními programy je v jejich určení. Pomocí aplikačních programů mohou uživatelé dělat některé užitečné věci (popří-padě se bavit – je-li aplikace, kterou si zrovna spustili, počítačová hra). Systémové programy jsou potřebné k tomu, aby systém vůbec fungoval. Textový editor je aplikace, mount je systémový pro-gram. Hranice mezi aplikačními a systémovými programy je často dost neostrá a je důležitá jen pro zásadové „kategorizéry“.
Součástí operačního systému mohou být i překladače programovacích jazyků a jejich knihovny (v případě Linuxu překladač GCC a knihovna jazyka C), nicméně tomu tak nemusí být. Další částí systému je dokumentace, někdy dokonce i některé hry. Tradičně se za operační systém pokládá obsah jeho instalační pásky či instalačních disků. Pokud jde o systém Linux, není uvedená defi-nice zcela jasná, protože na mnoha serverech FTP po celém světě existuje množství různých verzí systému.

Důležité části jádra systému

Jádro Linuxu sestává z několika důležitých subsystémů. Jsou to části řízení procesů, správy pamě-ti, ovladačů technických prostředků, ovladačů souborových systémů, správy sítě a různé další kusy a kousky. Některé z nich jsou zobrazeny na obrázku 2.1.

Snad nejdůležitějšími subsystémy (bez nichž nic jiného nefunguje) jsou správce paměti a správce procesů. Subsystém správy paměti zajišťuje přidělování paměťových oblastí a odkládacího prosto-ru jednotlivým procesům, částem jádra a vyrovnávací paměti. Subsystém správy procesů vytváří procesy a přepínáním mezi aktivními procesy, které využívají procesor, zabezpečuje multitasking.

Jádro systému na nejnižší úrovni obsahuje ovladače pro všechny druhy technických zařízení, které operační systém podporuje. Vzhledem k tomu, že na světě existuje celá řada různých typů hard-waru, je počet ovladačů zařízení velký. Je ale mnoho jinak podobných zařízení, která se často liší pouze v tom, jak spolupracují s programy. Takovéto podobnosti umožňují definovat obecné třídy ovladačů, jež podporují podobné operace. Každý člen takovéto třídy má stejné rozhraní k ostat-ním částem jádra. Liší se v tom, jak tyto operace implementuje. Například všechny ovladače disků vypadají pro zbytek jádra podobně. To znamená, že všechny znají operace jako „inicializuj dis-kovou jednotku“, „čti sektor N“ a „zapiš sektor N“.

Některé softwarové služby, jež poskytuje jádro samotné, mají rovněž podobné vlastnosti. Proto mohou být také rozdělené do tříd. Kupříkladu různé síťové protokoly byly vyčleněny do jednoho programového rozhraní – knihovny „BSD socket library“. Dalším příkladem je vrstva *virtuálního souborového systému* (VFS). Ta odděluje operace souborového systému od jejich implementace. Každý typ souborového systému obstarává implementaci určité množiny operací, společné všem systémům souborů. Když se některý z prvků systému pokouší využít určitý souborový systém, žádost jde přes VFS. Ten ji směruje k požadovanému ovladači konkrétního systému souborů.

Podrobnější diskuse o vnitřních funkcích jádra jsou uvedeny na této adrese: *http://www.tldp.org/LDP/lki/index.html*. Tento dokument je napsán pro jádro 2.4. Jakmile najdu popis jádra 2.6, uveřejním ho zde. Český překlad byl o některé informace vztahující se k jádru 2.6 doplněn.

Nejdůležitější služby v unixovém systému

Tato kapitola popisuje některé významnější služby systému Unix, avšak opět bez větších podrobností. Všechny služby budou později podrobně vysvětleny v dalších kapitolách.

Proces init

Nejdůležitější služby v systému Unix poskytuje proces init. Spuštění procesu init jako prvního z procesů je v každém unixovém systému posledním krokem, který provede jádro systému při zavádění. Po spuštění proces init pokračuje v proceduře zavádění systému. Vykonává různé úkoly, které se při spouštění systému obvykle provádí (kontroluje a připojuje souborové systémy, spouští démony a tak dále).

Přesný seznam úloh, které proces init při zavádění dělá, závisí na verzi tohoto programu i ope-račního systému. Proces init často obstarává takzvaný *jednouživatelský režim*. V jednouživatel-ském režimu se do systému nemůže nikdo přihlásit a příkazový interpret může z konzoly použí-vat pouze root. Běžným režimem práce je *víceuživatelský režim*. Tyto režimy práce některé systé-my Unix zobecňují do takzvaných *úrovní běhu* (*runlevel*). Jednouživatelský a víceuživatelský režim tak představují dvě různé úrovně, na kterých může systém běžet. Kromě nich mohou exis-tovat i další, například úroveň, při které se na konzole spustí grafické rozhraní X Window a podobně.

Linux povoluje až 10 úrovní běhu, 0-9, standardně však bývají definovány jen některé. Úroveň 0 představuje zastavení systému. Úroveň 1 je jednouživatelský režim. Úroveň 6 je restart systému. Chování dalších úrovní závisí na tom, jak je definuje vámi používaná distribuce, a v různých distri-bucích se výrazně liší. Podíváte-li se na soubor /etc/inittab, může vám napovědět, jaké jsou předdefinované úrovně a jak jsou nastaveny.

V běžné situaci proces init kontroluje, zda fungují procesy getty (umožňující uživatelům připojit se do systému), a adoptuje procesy – sirotky. Sirotci jsou procesy, jejichž rodičovské procesy byly z různých důvodů ukončeny – říká se, že umřely. V systému Unix *musí* být *všechny* procesy sou-částí jediné hierarchické stromové struktury. Proto musí proces init sirotky adoptovat.

Když se systém vypíná, proces init zodpovídá za ukončení všech ostatních procesů, odpojení všech souborových systémů, zastavení procesoru a za vše ostatní, co má podle dané konfigurace udělat.

Přihlášení z terminálů

Přihlášení uživatelů prostřednictvím terminálů (připojených na sériové linky) a konzoly (v přípa-dě, že neběží X Window) obstarává program getty. Proces init spouští zvláštní instanci getty pro každý terminál, ze kterého se bude možno do systému přihlásit. Program getty dále čte zadáva-né uživatelské jméno a spouští program login, jenž čte přístupové heslo. Jestli jsou uživatelské jméno a heslo správné, spustí program login příkazový interpret neboli *shell*. Když je příkazový interpret ukončen – jakmile se uživatel odhlásí ze systému nebo když je program login ukončen proto, že nesouhlasí uživatelské jméno a heslo – proces init to zjistí a spustí pro daný terminál novou instanci programu getty. Samotné jádro systému nemá vůbec přehled o přihlašování uži-vatelů do systému. Všechno kolem toho obstarávají systémové programy.

Syslog

Jádro systému i mnoho systémových programů hlásí různé chyby, vypisuje varování a jiná hláše-ní. Velmi často je důležité, aby bylo možno tyto zprávy prohlížet později, dokonce i s velkým časo-vým odstupem. Je tedy vhodné je zapisovat do nějakých souborů. Program, který to má na sta-rosti, se jmenuje syslog. Lze jej nastavit tak, aby třídil zprávy a hlášení do různých souborů, a to podle původce, případně stupně významnosti. Hlášení jádra systému jsou obvykle směrována do jiného souboru než hlášení jiných procesů a programů. Jsou většinou významnější a je potřeba číst je pravidelně, aby bylo možné rozeznat případné problémy v zárodku.

Periodické vykonávání příkazů: cron a at

Uživatelé i správci systému často potřebují spouštět některé programy pravidelně. Například admi-nistrátor systému, který musí

sledovat zaplněnost disku, by mohl chtít pravidelně spouštět příkaz, jenž by „vyčistil“ adresáře dočasných souborů (/tmp a /var/tmp). Program by odstranil starší dočasné soubory, které po sobě programy z různých důvodů korektně nesmazaly.

Takovéto služby nabízí program cron. Každý uživatel má vlastní soubor crontab, jenž obsahuje seznam příkazů, které chce vlastník spustit, a časy, kdy se mají tyto příkazy provést. Démon cron má na starosti spouštění těchto příkazů v požadovaném čase.
Služba at je podobná službě cron, provede se ale jenom jednou. Příkaz je vykonán v určenémčase, ale jeho spouštění se neopakuje.Podrobnější informace viz manuálové stránky cron(1), crontab(1), crontab(5), at(1) a atd(8).

Grafické uživatelské rozhraní

Unix a Linux nezačleňují uživatelská rozhraní do jádra systému. Místo toho je implementují pomo-cí programů uživatelské úrovně. To se týká jak textového módu, tak grafického uživatelského pro-středí. Díky takovémuto řešení je samotný systém flexibilnější. Má to ale nevýhodu v tom, že je na druhou stranu velmi jednoduché implementovat pro každý program různá uživatelská rozhra-ní. Důsledkem je, že se takovýto systém uživatelé pomaleji učí.

Grafické prostředí, které Linux používá primárně, se nazývá „X Window System“ (zkráceně X nebo X11). Ale samotný systém X Window ještě neznamená uživatelské rozhraní. X Window pouze implementuje systém oken, tedy sadu nástrojů, pomocí kterých může být grafické uživatelské roz-hraní implementované. Nad tímto systémem pracují správci oken, populární jsou například fvwm, icewm, blackbox a windowmaker. Dále existují dva rozšíření správci pracovního prostředí – Gnome a KDE.

Komunikace prostřednictvím počítačové sítě

Komunikace pomocí počítačové sítě je propojení dvou nebo více počítačů tak, že mohou komunikovat navzájem každý s každým. V současnosti používané metody propojování a komunikace jsou docela komplikované, ale výsledný efekt stojí za to.

Operační systémy Unix mají řadu síťových funkcí. Většinu základních služeb – služby souboro-vých systémů, tisky, zálohování a podobně – lze využívat i prostřednictvím sítě. To ulehčuje sprá-vu systému a umožňuje centralizovanou administraci. Zachovávají se výhody mikropočítačové technologie i přínos distribuovaných systémů (nižší náklady a lepší odolnost vůči poruchám). Tato kniha se komunikací prostřednictvím počítačové sítě zabývá jenom zběžně. Podrobnosti
o této problematice, včetně základního popisu principů počítačových sítí, přináší *Příručka správ-ce sítě* na adrese *http://www.tldp.org/LDP/nag2/index.html*.

Přihlášení do systému ze sítě

Přihlášení do systému ze sítě funguje trochu odlišně než běžné přihlášení přes terminál. Pro každý terminál, prostřednictvím kterého je možné se přihlásit, je vyhrazená samostatná fyzická sériová linka. Pro každého uživatele, který se přihlašuje prostřednictvím sítě, existuje jedno samostatné vir-tuální síťové spojení, nicméně těchto spojení může být velký počet. Proto není možné, aby běže-ly samostatné procesy getty pro všechna možná virtuální spojení. Kromě toho existuje několik různých způsobů přihlášení prostřednictvím sítě. Dva nejdůležitější způsoby v sítích TCP/IP jsou telnet a ssh.

Síťová přihlášení mají místo řady procesů getty jednoho démona pro každý ze způsobů připojení (telnet a rlogin mají každý vlastního démona). Tento démon vyřizuje všechny přicházející žádos-ti o přihlášení. Dostane-li takovouto žádost, spustí svou novou instanci. Nová instance pak obslu-huje tuto jedinou žádost a původní instance nadále sleduje další příchozí žádosti o přihlášení. Nová instance pracuje podobně jako program getty.

Síové souborové systémy

Jednou z nejužitečnějších věcí, kterou lze využít díky síťovým službám, je sdílení souborů pomo-cí *síťového souborového systému*. V závislosti na konkrétní síti lze využívat Network File System (NFS) nebo Common Internet File System (CIFS). Služba NFS je běžná na systémech UNIX, zatím-co CIFS patří spíše k Windows. V Linuxu je NFS podporovaná jádrem, CIFS však nikoli. CIFS implementuje projekt Samba: *http://www.samba.org*.

Operace provedené programem na jednom počítači jsou prostřednictvím síťového souborového systému poslány na jiný počítač. Program tak má dojem, že soubory na jiném počítači jsou na tom počítači, na němž program běží. Sdílení informací se tak zcela zjednodušuje, neboť není nutná úprava programů.

Podrobnosti najdete v kapitole „NAS (Network Attached Storage)“.

Pošta

Elektronická pošta je obvykle tím nejdůležitějším způsobem počítačové komunikace. Elektronický dopis je uložený v textovém souboru se zvláštním formátem. K jeho odeslání nebo přečtení se používají speciální programy.

Každý uživatel systému má vlastní *schránku na příchozí poštu*. Je to soubor určitého formátu, ve kterém jsou uloženy všechny nově příchozí zprávy. Když někdo odesílá poštu, program zjistí adre-su poštovní schránky příjemce a připojí dopis k jeho souboru

s příchozí poštou. Jestli je schrán-ka příjemce na jiném počítači, je dopis odeslán na tento stroj a ten se bude snažit doručit jej do schránky příjemce. Systém elektronické pošty se skládá z několika typů programů. Doručení pošty do místních nebo vzdálených poštovních schránek má na starosti první z nich – *agent pro přenos pošty* (MTA), např-klad sendmail nebo postfix. Uživatelé používají ke čtení pošty množství různých programů, tak-zvaných *uživatelských poštovních agentů* (MUA), například pine nebo evolution. Poštovní schránky uživatelů jsou obvykle uloženy v adresáři /var/spool/mail, pokud si je MUA neuloží jinam.

Další informace o nastavování a provozování poštovních služeb si můžete přečíst v Návodu ke správě pošty (Mail Administrator HOWTO) na *http://www.tldp.org/HOWTO/Mail-Administrator-HOWTO.html* nebo navštívit stránky programů sendmail a postfix: *http://www.sendmail.org/*, resp. *http://www.postfix.org/*.

Tisk

Tiskárnu může současně využívat pouze jeden uživatel. Nesdílet tiskárny mezi uživateli je ale dost neekonomické. Tiskárnu proto řídí program, jenž realizuje takzvanou *tiskovou frontu*. Všechny tis-kové úlohy všech uživatelů systému jsou zařazeny do fronty. Hned, jak tiskárna ukončí jednu úlohu, automaticky se jí odesílá další v pořadí. Uživatelé si nemusí zabezpečovat frontu požadav-ků na tisk organizačně a odpadá i nutnost soupeřit a handrkovat se o přístup k tiskárně. Místo toho vytvářejí frontu na tiskárnu, v níž čekají na vyřízení jednotlivé výstupy, aniž by uživatelé věděli, kdy skutečně skončil jejich tisk. To je bezesporu značný přínos k uspořádaným mezilid-ským vztahům na pracovišti.

Program pro obsluhu tiskové fronty navíc *ukládá* (tzv. *spool*) všechny tiskové výstupy na disk, takže pokud je tisková úloha ve frontě, je text uložen v nějakém souboru. Tento mechanismus aplikačním programům umožňuje rychle odeslat tiskové úlohy programu, jenž tiskovou frontu obsluhuje. Aplikace sama tak může pokračovat ve své práci. Nemusí čekat, než se úloha, která se právě tiskne, ukončí. To je v mnoha případech skutečně výhodné. Umožní vám to například zahá-jit tisk jedné verze dokumentu, přičemž nemusíte čekat, než se tisk ukončí, a můžete mezitím pra-covat na nové, zcela pozměněné verzi.

Podrobný návod k nastavování tiskáren naleznete v páté části knihy v kapitole „Tisk v Linuxu“.

Organizace systému souborů

Souborový systém je rozdělen na mnoho částí. Obvykle jsou hierarchicky uspořádané a nejvýše stojí kořenový souborový systém root. Souborový systém root obsahuje adresáře /bin, /lib, /etc, /dev a několik dalších; souborový systém /usr obsahuje programy a data, jež se nemění. Souborový systém /var obsahuje data, jež se naopak často mění (například logovací soubory), a souborový systém /home obsahje osobní soubory. Rozdělení může být v závislosti na hardwaru a rozhodnutí správce systému zcela jiné; dokonce mohou být všechny soubory v jediném soubo-rovém systému.

V kapitole „Přehled adresářové struktury“ je uspořádání souborového systému popsáno poněkud podrobněji; ještě detailnější popis naleznete v dokumentu Filesystem Hierarchy Standard na adre-se *http://www.pathname.com/fhs/*.

Přehled adresářové struktury

Dva dny nato seděl Pú na své větvi, pohupoval nohama a tam, vedle něj, stály čtyři hrníčky medu...

A. A. Milne

Kapitola popisuje důležité části standardní struktury adresářů operačního systému Linux, která je založená na standardu Filesystem Hierarchy Standard. Načrtneme v ní také běžný způsob rozdě-lení struktury adresářů do samostatných souborových systémů (svazků) s odlišnými účely a tento způsob rozdělení zdůvodníme. Ne všechny distribuce Linuxu tento standard dodržují, je však dostatečně univerzální, aby stačil pro základní přehled.

Pozadí

Tato kapitola volně vychází z normy Filesystem Hierarchy Standard (FHS) verze 2.1, která je poku-sem zavést jisté konvence do organizace adresářového stromu operačního systému Linux8. Výho-dou přijetí takovéto normy je, že když bude vše na svém obvyklém místě, bude jednodušší psát pro-gramy a přenášet na Linux software z jiných platforem. Zároveň to ulehčí správu

počítačů, na kte-rých běží operační systém Linux. I když neexistuje autorita, která by vývojáře, programátory a distri-butory donutila přizpůsobit se této normě, je její podpora v současnosti součástí většiny (ne -li všech) distribucí Linuxu. Není vhodné se neřídit standardem FHS, nejsou-li pro to velmi závažné důvody. Norma FHS se snaží sledovat tradice Unixu i současné trendy jeho vývoje. Motivací je snaha o usnad-nění přechodu na Linux pro uživatele, kteří mají zkušenosti s jinými systémy Unix a naopak.

Tato kapitola není tak detailní jako samotná norma FHS. Správce systému – chce-li plně pronik
nout do problematiky systémů souborů – by si měl přečíst i normu FHS. Kapitola rovněž nepopisuje podrobnosti týkající se všech typů souborových systémů. Nebylo také cílem popsat všechny adresáře a soubory, ale nabídnout čtenáři přehled o celém systému z per-spektivy systému souborů. Podrobnější informace o popisovaných souborech jsou k dispozici na jiných místech této knihy, případně na manuálových stránkách.

Celou stromovou strukturu adresářů je možné rozdělit na menší části, každá z těchto částí může být umístěna na vlastním disku nebo samostatné diskové oblasti. Tak se lze jednoduše přizpůso-bit omezením velikosti disků a zároveň usnadnit zálohování i ostatní úkoly spojené se správou systému.
Nejdůležitější z těchto částí jsou kořenový souborový systém a dále souborové systémy /usr, /var a /home (viz obrázek 3.1). Každý systém souborů má jiné určení. Adresářová stromová struktura byla navržena tak, aby fungovala i v síti počítačů s operačním systémem Linux. Uživatelé a pro-gramy tak mohou pomocí sítě sdílet některé části systémů souborů, a to buď prostřednictvím zaří-zení určených pouze pro čtení (například CD-ROM) nebo pomocí sítě se systémem NFS.

Smysl jednotlivých částí adresářové struktury je popsán v dalším textu.

Kořenový souborový systém je specifický pro každý počítač. Obecně je uložen na lokál-ním disku (avšak může to být i virtuální disk v paměti RAM – takzvaný „ramdisk“ nebo síťo-vá disková jednotka). Kořenový svazek obsahuje soubory nutné pro zavedení systému a jeho uvedení do stavu, ve kterém mohou být připojeny ostatní souborové systémy. Obsah kořenového souborového systému postačuje pro práci v jednouživatelském režimu. Na tomto svazku jsou rovněž uloženy nástroje pro opravy poškozeného souborového systému a pro obnovení ztracených souborů ze záloh.
Souborový systém /usr obsahuje všechny příkazy, knihovny, manuálové stránky a jiné soubory, jejichž obsah se nemění a které uživatel potřebuje při běžném provozu. Žádný ze souborů na tomto svazku by neměl být specifický pro daný počítač. Rovněž by se neměl při normálním provozu měnit. Tyto podmínky zaručují, že soubory uložené v souborovém systému /usr bude možné efektivně sdílet v síti. Sdílení tohoto svazku je výhodné jak z hlediska nákladů – šetří se tím místo na disku (v souborovém systému /usr mohou být uloženy stovky megabajtů dat) – tak z hlediska usnadnění správy systému (například při instalaci novější verze aplikace se pak mění pouze systém /usr na hostitelském počítači a ne na každé stanici zvlášť). Je-li souborový systém /usr na lokálním disku, může být při-pojen pouze pro čtení. To snižuje pravděpodobnost poškození systému souborů při havá-rii systému.
Souborový systém /var obsahuje soubory, které se v čase mění. Tedy především sdílené adresáře pro elektronickou poštu, systém news, tiskárny, logovací soubory, formátované manuálové stránky a dočasné soubory. Historicky bývaly všechny soubory, které jsou nyní uloženy v systému /var, v souborovém systému /usr. To však znemožňovalo připojit sva-zek /usr pouze pro čtení.
Souborový systém /home obsahuje domovské adresáře uživatelů, tedy všechna „reálná data“. Vyčlenění uživatelských domovských adresářů do vlastní adresářové stromové struk-tury nebo samostatného souborového systému ulehčuje zálohování. Ostatní části adresá-řového stromu totiž buď nevyžadují zálohování vůbec nebo – vzhledem k tomu, že se nemění tak často – se zálohují jenom zřídka. Velký souborový systém /home je dobré roz-dělit na několik menších částí hierarchicky nižší úrovně a rozlišit je jménem, například /home/students a /home/staff.

Různé části, na které je hierarchická adresářová struktura rozčleněna, byly v našem přehledu ozna-čeny jako souborové systémy. Není ale žádný zvláštní důvod k tomu, aby ve skutečnosti ležely na samostatných oddělených svazcích. Všechny by mohly být nakonec i v jediném souborovém systé-mu. Takové řešení má význam hlavně pro malé jednouživatelské systémy, kdy je prioritou jedno-duchost.

Celá stromová adresářová struktura může být rozdělena na souborové systémy i jinak. Způsob její-ho rozčlenění závisí na tom, jak velký je disk a jak velký diskový prostor bude vyhrazen pro různé účely. Jedinou věcí, na kterou je potřeba dbát, jsou standardní unixová *jména*. Ta je potřeba zachovat. I když bude adresář /var a /usr ve stejné diskové oblasti, musí být zachována stan-dardní jména, jako např. /usr/lib/libc.a nebo /var/log/messages. Totéž platí i v případě, že se například přesune adresář /var do adresáře /usr/var a na původním místě přesunutého adre-sáře bude symbolický odkaz na adresář /usr/var.

Struktura souborového systému Unixu sdružuje soubory podle jejich účelu, tedy všechny příkazy na jednom místě, data na jiném, dokumentace na dalším a tak dál. Alternativou by bylo sdružo-vat soubory podle toho, ke kterému programu patří. Pak by mohly být například všechny soubo-ry pro program Emacs v jednom adresáři, všechny soubory pro TeX v jiném a podobně. Problém druhého přístupu je v tom, že je velmi obtížné sdílet soubory (adresář určitého programu často obsahuje jak statické soubory, které lze sdílet, tak soubory, jejichž obsah se mění, a ty sdílet nelze). Rovněž by bylo velmi složité dohledávat v rámci celého systému soubory určitého typu, například manuálové stránky aplikací uložené na mnoha různých místech. Programátory by jistě strašila noční můra – jak v takovémto případě vytvořit programy, které by byly schopné v případě potře-by manuálové stránky všech aplikací nalézt.

Souborový systém root

Kořenový svazek – *root* – by obecně měl být malý, protože obsahuje velmi kritické soubory. U malého souborového systému, který se mění jenom zřídka, je menší pravděpodobnost poško-zení. Poškození souborového systému root většinou znamená, že operační systém nebude možné zavést. Tento problém lze řešit pouze pomocí speciálních opatření (například zavedením systému z diskety), a to by chtěl riskovat asi málokdo.

Obecně by kořenový adresář neměl obsahovat žádné soubory, snad kromě standardního obrazu systému. Ten se obvykle jmenuje /vmlinuz (v distribucích je velmi často umístěn v podadresáři /boot). Všechny ostatní soubory by měly být uloženy v podadresářích kořenového adresáře, obvykle tímto způsobem:

/bin

Příkazy potřebné při zavádění systému, které mohou použít i normální uživatelé.

/sbin Stejně jako /bin, příkazy ale nejsou určeny pro normální uživatele, i když i ti je mohou pou-žít, je-li to nutné a je-li to povoleno. Tento adresář se obvykle nenachází v cestě normálních uživatelů, má jej ale v cestě root.

/etc

Konfigurační soubory specifické pro daný počítač.

/root

Domovský adresář superuživatele, obvykle nepřístupný ostatním uživatelům.

/lib

Sdílené knihovny pro programy v kořenovém souborovém systému.

/lib/modules

Zaveditelné moduly jádra systému – zvláště ty, které jsou potřeba pro zavedení systému při zotavení po havárii (například síťové ovladače a ovladače pro souborový systém).

/dev

Soubory zařízení. Speciální soubory, které vytvářejí uživatelská rozhraní pro různá zařízení v systému.

/tmp

Dočasné soubory. Jak název napovídá, běžící programy sem ukládají dočasné soubory.

/boot

Soubory, jež používá zavaděč operačního systému, například LILO nebo GRUB. Často se zde ukládají obrazy jádra (místo v kořenovém adresáři). V případě, že jich máte víc, může obsah adresáře /boot značně narůst a pak bude lepší mít jej v samostatném souborovém systému. Tím se také zajistí, že obrazy jádra budou uloženy na prvních 1 024 cylindrech disku IDE. Omezení na prvních 1 024 cylindrů dnes už většinou neplatí. Moderní BIOSy a nové verze LILO umožňují tento limit obejít prostřednictvím logického adresování bloků (LBA). Viz manu-álová stránka programu lilo.

/mnt

Přípojné místo pro dočasná připojení dalších systémů souborů správcem systému. Nepředpo-kládá se, že by tento adresář využívaly pro automatická připojení souborových systémů pro-gramy. Adresář /mnt může být rozdělen na podadresáře (například /mnt/dosa pro disketovou mechaniku používanou v souborovém systému MS-DOS a /mnt/exta pro tutéž mechaniku využívanou se souborovým systémem ext2 a podobně). Nové verze distribucí používají v sou-ladu s novou specifikací FHS pro vyměnitelná média adresář /media (diskety, CD-ROM apod.).

/proc, /usr, /var, /home Přípojná místa pro další souborové systémy. Konkrétně systém /proc ovšem fyzicky na disku neexistuje, viz kapitolu věnovanou tomuto souborovému systému.

Adresář /etc

Adresář /etc obsahuje mnoho souborů. Některé z nich jsou popsány v dalším textu. Pokud jde o ostatní, měli byste nejdřív zjistit, ke kterému programu patří, a pak si přečíst manuálové strán-ky k tomuto programu. V adresáři /etc je uloženo také hodně síťových konfiguračních souborů, které jsou popsány v *Příručce správce sítě*.

/etc/rc nebo /etc/rc.d nebo /etc/rc?.d Skripty nebo adresáře skriptů, které se spouští při startu nebo v případě, že se mění úroveň běhu systému. Podrobnosti viz kapitola „Proces init“.

/etc/passwd

Databáze uživatelů systému s položkami, v nichž je uloženo uživatelské jméno i skutečné jméno uživatele, domovský adresář, šifrované heslo a některé další informace. Formát je popsán v manuálové stránce programu passwd.

/etc/shadow

Tento soubor je zašifrovaný a obsahuje uživatelská hesla.

/etc/fdprm

Tabulka parametrů disketové jednotky. Popisuje, jak vypadají různé formáty disket. Používá ji program setfdprm. Víc informací uvádí manuálová stránka programu setfdprm.

/etc/fstab

Seznamy souborových systémů připojovaných automaticky při startu příkazem mount -a (ve skriptu /etc/rc nebo nějakém ekvivalentu). V systému Linux obsahuje rovněž informace o odkládacích oblastech, které používá příkaz swapon -a. Podrobnější informace obsahuje kapitola „Připojení a odpojení“ a manuálové stránky příkazu mount. Samotný soubor fstab má typicky svou vlastní manuálovou stránku v sekci 5.

/etc/group Soubor podobný souboru /etc/passwd, ale místo uživatelů popisuje pracovní skupiny. Podrobnější informace viz manuálová stránka group.

/etc/inittab

Konfigurační soubor procesu init.

/etc/issue

Soubor obsahuje výstup programu getty, který se zobrazí před výzvou pro přihlášení uživate-le. Obvykle obsahuje stručný popis systému nebo uvítací hlášení. Obsah určuje správce syst-mu.

/etc/magic

Konfigurační soubor programu file. Obsahuje popisy různých formátů souborů, podle kterých pak program file tyto typy rozpoznává. Více informací najdete v manuálových stránkách pro magic a file.

/etc/motd

Takzvaná *zpráva pro tento den* – automatický výstup na terminál uživatele po úspěšném při-hlášení do systému. Obsah volí správce systému. Často se využívá pro předávání informací (například upozornění na plánovaná zastavení systému apod.) všem uživatelům systému.

/etc/mtab

Seznam aktuálně připojených souborových systémů. Jeho obsah po zavedení systému a při-pojení určených souborových systémů prvotně nastavují inicializační skripty, v běžném pro-vozu pak automaticky příkaz mount. Používá se v případech, kdy je potřeba zjistit, které sou-borové systémy jsou připojené, například při zadání příkazu df.

/etc/login.defs

Konfigurační soubor příkazu login. Obvykle bývá popsán v sekci 5 manuálu.

/etc/printcap Podobně jako u souboru /etc/termcap, až na to, že soubor je určený pro tiskárny. Odlišná je i jeho syntaxe. Bývá popsán v sekci 5 manuálových stránek. /etc/profile, /etc/bacsh.rc, /etc/csh.cshrc Soubory spouštěné při přihlášení uživatele nebo při startu systému interprety příkazů Bourne BASH a C shell. Umožňují správci systému stanovit globální nastavení stejná pro všechny uži-vatele. Viz manuálové stránky k příslušným interpretům příkazů. Uživatelé si také mohou vytvářet vlastní kopie těchto adresářů ve svých domovských adresářích, aby si zálohovali pro-středí, ve kterém pracují.

/etc/securetty

Soubor identifikuje zabezpečené terminály, tedy terminály, ze kterých se může přihlašovat superuživatel. Typicky jsou v seznamu uvedeny pouze virtuální konzoly, takže je nemožné (nebo přinejmenším těžší) získat oprávnění superuživatele přihlášením se po modemu nebo ze sítě. Nepovolujte přihlášení superuživatele přes síť. Rozumnější je přihlásit se jako běžný uživatel a pak získat práva superuživatele příkazem su nebo sudo.

/etc/shells

Soubor, jenž uvádí seznam důvěryhodných interpretů příkazů. Příkaz chsh umožňuje uživa-telům změnit příkazový interpret spuštěný při přihlášení, a to pouze na některý z interpretů uvedený v tomto souboru. Proces ftpd, který běží na hostitelském počítači a poskytuje služ-by FTP pro klientské počítače, rovněž kontroluje, zda je uživatelův příkazový interpret uveden v tomto souboru, a nedovolí připojit se klientům, jejichž příkazový interpret v tomto seznamu uveden není.

/etc/termcap

Databáze vlastností terminálů. Popisuje, kterými escape sekvencemi se řídí různé typy termi-nálů. Každý program je napsaný tak, že místo přímého výstupu escape sekvence, jež by fun-govala pouze s konkrétním typem terminálu, hledá v tabulce /etc/termcap sekvenci, která odpovídá tomu, co chce program na terminálu zobrazit. Pak může většina programů správně obsluhovat většinu typů terminálů. Více informací uvádí manuálové stránky pro termcap, curs-termcap a terminfo.

Adresář /dev

Adresář /dev obsahuje speciální soubory všech zařízení. Speciální soubory se vytváří v průběhu instalace operačního systému, v běžném provozu pak skriptem /dev/MAKEDEV. Podobný je skript /dev/MAKEDEV.local. Ten upravuje a používá správce systému, když vytváří čistě lokální speci-ální soubory a odkazy. Lokální speciální soubory jsou ty, které nejsou vytvořeny standardním postupem pomocí skriptu MAKEDEV, typicky například speciální soubory pro některé nestan-dardní ovladače zařízení.

Následující seznam není v žádném případě vyčerpávající nebo tak podrobný, jak by měl být. Mnohé z těchto souborů zařízení budou potřebovat podporu, kterou je nutno přeložit společně s jádrem. Podrobnosti k jednotlivým zařízením viz dokumentace jádra. S novou verzí jádra 2.6 – a tudíž i v nových distribucích – se soubory zařízení tvoří dynamicky podle potřeby (např. při při-pojení zařízení). Pro vytváření souborů se používá démon udev, více informací najdete v jeho manuálových stránkách. Domníváte-li se, že některá důležitá zařízení chybějí, dejte mi, prosím, vědět. Pokusím se je doplnit do příští revize.

/dev/dsp

Digital Signal Processor. Tvoří rozhraní mezi softwarem zpracovávajícím zvuk a zvukovou kar-tou. Je to znakové zařízení na hlavním čísle 14 a vedlejším 3. V novém jádře s ALSA ovladači přebírají jeho funkci zařízení v adresáři /dev/snd/, konkrétně zařízení pcmXY.

/dev/fd0

První disketová mechanika. Máte-li to štěstí, že jich máte několik, budou číslovány postupně. Je to znakové zařízení na hlavním čísle 2 a vedlejším 0.

/dev/fb0

První zařízení framebufferu. Framebuffer je abstraktní vrstva mezi softwarem a grafickým hard-warem. To znamená, že aplikace nemusí nic vědět o konkrétním hardwaru, pouze komuni-kuje s rozhraním ovladače rámcového bufferu (API, Application Programming Interface), které je dostatečně definované a standardizované. Znakové zařízení pro framebuffer je na hlavním čísle 29 a vedlejším 0.

/dev/hda /dev/hda je hlavní (master) mechanika IDE na primárním řadiči IDE, /dev/hdb je pomocná (slave) mechanika na primárním řadiči, /dev/hdc a /dev/hdd jsou hlavní (resp. vedlejší) zaří-zení na sekundárním řadiči (případně na druhém kanále řadiče EIDE). Každý disk je rozdělen na oblasti (partitions). Oblasti 1–4 jsou primární a oblasti 5 a vyšší jsou logické oblasti uvnitř rozšířené oblasti. Proto se soubor, který se odkazuje na oblasti, skládá z několika částí. Napří-klad /dev/hdc9 odpovídá oblasti 9 (logická oblast uvnitř rozšířené oblasti) na hlavní mecha-nice IDE sekundárního řadiče IDE. Hlavní a vedlejší čísla jsou složená. Pro první řadič IDE jsou všechny oblasti bloková zařízení na hlavním čísle 3. Hlavní mechanika hda je na vedlejším čísle 0 a pomocná mechanika hdb je na vedlejším čísle 64. Ke každé oblasti uvnitř mechaniky při-čteme číslo oblasti k vedlejšímu číslu mechaniky. Například /dev/hdb5 je hlavní 3, vedlejší 69 (64 + 5 = 69). U mechanik na sekundárním rozhraní se postupuje stejně, pouze s hlavním čís-lem 22.

/dev/ht0 První mechanika pásky IDE. Následující mechaniky jsou číslovány ht1 atd. Jsou to znaková zařízení na hlavním čísle 37 a začínají na vedlejším čísle 0 pro ht0, 1 pro ht1 atd.

/dev/js0 První analogový joystick. Následující joysticky jsou číslovány js1, js2 atd. Digitální joysticky se nazývají djs0, djs1 atd. Jsou to znaková zařízení na hlavním čísle 15. Analogové joysticky začínají na vedlejším čísle 0 a jdou až k 127 (víc než dost i pro nejfantastičtějšího hráče). Digi-tální joysticky začínají na vedlejším čísle 128.

/dev/lp0 První paralelní tiskárna. Následující tiskárny jsou číslovány lp1, lp2 atd. Jsou to znaková zaří-zení na hlavním čísle 6 a vedlejších číslech od 0 a dále sekvenčně.

/dev/loop0

První tzv. *loopback* zařízení. Tato zařízení se používají na připojení souborových systémů, které nejsou umístěny na jiných blokových zařízeních typu disk. Když například chcete při-pojit obraz CD ROM iso9660 bez vypalování do CD, musíte použít loopback. To je obvykle transparentní vzhledem k uživateli a provádí se to příkazem mount. Viz manuálové stránky příkazů mount a losetup. Loopback zařízení jsou bloková zařízení na hlavním čísle 7 a s ved-lejšími čísly začínajícími na 0 a dále číslovány sekvenčně.

/dev/md0

První skupina metadisků. Metadisky se vztahují k zařízením RAID (Redundant Array of Inde-pendent Discks). Další podrobnosti viz nejnovější návod RAID HOWTO a LDP, které nalez-nete na *http://www.tldp.org/HOWTO/Software-RAID-HOWTO.html.* Metadisková zařízení jsou bloková zařízení na hlavním čísle 9 s vedlejšími čísly začínajícími na 0 a dále číslovány sekvenčně.

/dev/mixer

Součást ovladače OSS (Open Sound System). Podrobnosti viz dokumentace na adrese *http://www.opensound.com.* Jde o znakové zařízení na hlavním čísle 14 a vedlejším 0. I u něj platí, že v novém jádře s ALSA ovladači přebírají jeho funkci jiná zařízení.

/dev/null

Koš na bity. Černá díra, kam můžete posílat data, o něž už nemáte zájem. Vše, co pošlete na /dev/null, zmizí. Používá se, například když chcete provádět příkaz, jehož výstup se nemá objevovat na terminále. Jde o znakové zařízení na hlavním čísle 1 a vedlejším 3.

/dev/psaux

Port pro myš PS/2. Jde o znakové zařízení na hlavním číslu 10 a vedlejším 1.

/dev/pda

IDE disky připojené přes paralelní port. Pojmenování je podobné jako u interních řadičů IDE (/dev/hd*). Jsou to bloková zařízení na hlavním čísle 45. Vedlejší čísla vyžadují trochu důklad-nější vysvětlení. První zařízení je /dev/pda a je na vedlejším čísle 0. Oblasti na tomto zaříze-ní se vyhledávají přidáním čísla oblasti k vedlejšímu číslu zařízení. Každé zařízení může mít 15 oblastí (nikoli 63, jak je tomu u interních IDE disků). Vedlejší čísla /dev/pdb začínají na 16, /dev/pdc na 32 a /dev/pdd na 48. Takže například vedlejší číslo /dev/pdc6 bude 38 (32 + 6 = 38). Toto schéma poskytuje možnost mít 4 paralelní disky po 15 oblastech.

/dev/pcd0

CD ROM mechaniky připojené přes paralelní port. Jsou číslovány vzestupně od 0. Všechny jsou bloková zařízení na hlavním čísle 46, /dev/pcd0 je na vedlejším čísle 0, následují mecha-niky jsou na 1, 2, 3 atd.

/dev/pt0

Paralelní port pásky. Pásky nemají oblasti, a jsou tedy číslovány sekvenčně. Jsou to znaková zařízení na hlavním čísle 96. Vedlejší čísla začínají od 0 (/dev/pt0), dále 1 (/dev/pt1) atd.

/dev/parport0

Paralelní port bez určení. Většina zařízení připojených k paralelním portům má své vlastní ovladače. Toto je zařízení s přímým přístupem k portu. Je to znakové zařízení na hlavním číslu 99 s vedlejším číslem 0. Následující zařízení po prvním jsou číslována přičítáním jedné k vedlejšímu číslu.

/dev/random nebo /dev/urandom Oba soubory jsou generátory náhodných čísel v jádru. Soubor /dev/random je nedeterminis-tický generátor, což znamená, že hodnotu následujícího čísla nelze odvodit z čísla předchozí-ho. Ke generování čísel využívá entropii systémového hardwaru. Když ji vyčerpá, musí počkat na další a do té doby z něho nelze přečíst žádné číslo. Soubor /dev/urandom pracuje podob-ně. Nejprve také využívá entropii systémového hardwaru, avšak když ji vyčerpá, poskytuje pseudonáhodná čísla, která generuje pomocí příslušného vzorce. To se z hlediska bezpečnos-ti životně důležitých funkcí (např. generování dvojice klíčů) považuje za méně bezpečné.
Pokud je bezpečnost převažujícím hlediskem, používejte raději /dev/random. Dáváte-li před-nost rychlosti, zcela postačí /dev/urandom. Obě jsou znaková zařízení na hlavním čísle 1 s vedlejšími čísly 8 pro /dev/random a 9 pro /dev/urandom.

/dev/sda

První mechanika SCSI na první sběrnici SCSI, případně na SATA. Následující mechaniky jsou pojmenovány podobně jako v případě mechanik IDE: /dev/sdb je druhá mechanika SCSI, /dev/sdc je třetí mechanika atd.

/dev/ttyS0

První sériový port. Nejčastěji jej používáme k připojení externího modemu k systému.

/dev/zero

Jednoduchý způsob, jak získat mnoho nul (binárních). Výsledkem čtení z tohoto zařízení je vždy 0. To se může hodit například pro soubory s pevnou délkou, na jejichž obsahu nám nezá-leží. Je to znakové zařízení na hlavním čísle 1 a vedlejším 5.

Souborový systém /usr

Souborový systém /usr je často dost velký, protože jsou v něm instalované všechny programy. Všechny soubory v systému /usr jsou obvykle instalované přímo z distribuce systému Linux. Všechny další lokálně (myšleno pro každý počítač) instalované programy se ukládají do adresáře /usr/local. Tento adresář obvykle obsahuje software instalovaný jinak než z distribučních zdro-jů a jeho umístěním mimo standardní hierarchii v /usr je zabráněno případným kolizím či nechtě-ným aktualizacím. Některé z podadresářů adresáře /usr jsou popsány níže, ty méně významné neuvádíme. Více informací přináší popis standardu FHS.

/usr/X11R6

Všechny soubory systému X Window. Soubory pro X nejsou integrální součástí operačního systému z důvodů zjednodušení vývoje a instalace X. Adresářová struktura /usr/X11R6 je podobná stromu, který je vytvořen pod adresářem /usr samotným.

/usr/bin

Zde se nachází téměř všechny uživatelské příkazy. Některé další příkazy jsou uloženy v adre-
sáři /bin nebo /usr/local/bin. /usr/sbin

Obsahuje ty příkazy pro správu systému, které nejsou potřeba přímo v kořenovém souboro-vém systému (například převážná většina serverových programů). /usr/share/man, /usr/share/info, /usr/share/doc

Manuálové stránky, informační dokumenty GNU, případně různé jiné soubory s dokumentací.

/usr/include

Hlavičkové soubory pro programovací jazyk C. Z důvodů zachování konzistence by měly být spíš v adresáři /usr/lib, ale z historických důvodů jsou umístěny ve zvláštním adresáři. /usr/lib

Datové soubory pro programy a subsystémy, které se nemění. Jsou zde rovněž uloženy někte-ré globální konfigurační soubory. Jméno lib je odvozeno od anglického slova „library“ (kni-hovna). Původně totiž byly v adresáři /usr/lib uloženy knihovny podprogramů.

/usr/local

Místo pro lokálně instalovaný software a další soubory. Originální distribuce by zde neměla nic instalovat, tento adresář slouží čistě správci systému. Díky tomu si může být jistý, že aktualizace distribuce mu nepřepíší žádné jím instalované programy.

Souborový systém /var

Systém /var obsahuje data, která se při běžném provozu systému mění. Soubory jsou specifické pro každý systém, a proto se data mezi jinými počítači v síti nesdílí.

/var/cache/man

Vyrovnávací paměť pro manuálové stránky, které jsou formátovány na požádání. Zdrojové texty manuálových stránek jsou obvykle uloženy v adresáři /usr/share/man/man? (kde ? je příslušná sekce manuálu, viz manuál pro příkaz man, sekce 7). Některé stránky se dodávají v předem formátované verzi a jsou pak uloženy v adresáři /usr/share/man/cat*. Jiné strán-ky je třeba při prvním prohlížení naformátovat. Formátované verze jsou pak uloženy právě v adresáři /var/cache/man. Další uživatel, který si chce stejné stránky prohlížet, tak nemusí čekat na jejich opakované formátování.

/var/games Jakákoliv proměnná data her instalovaných v adresáři /usr – pro případ, že by byl svazek /usr připojen jen pro čtení.

/var/lib

Soubory, které se při normálním provozu systému mění.

/var/local Měnící se data pro programy instalované v adresáři /usr/local (tedy programy instalované správcem systému). Upozorňujeme, že takto instalované programy by měly podle potřeby používat i ostatní podadresáře nadřazeného adresáře /var, například /var/lock.

/var/lock Soubory zámků. Většina programů dodržuje určitou konvenci a vytváří v adresáři /var/lock zámky. Tím dávají ostatním programům najevo, že dočasně využívají některé zařízení nebo soubor. Jiné programy, které by chtěly stejné zařízení či soubor ve stejném okamžiku použí-vat, se o to nebudou pokoušet.

/var/log

Adresář obsahuje logovací soubory různých programů, zejména programu login (do souboru /var/log/wtmp se zaznamenávají všechna přihlášení a odhlášení uživatelů systému) a syslog (do souboru /var/log/messages ukládá všechna hlášení jádra systému a systémových pro-gramů). Velikost souborů v adresáři /var/log dost často nekontrolovaně roste, proto se musí v pravidelných intervalech mazat.

/var/mail

Standardem FHS navrhované umístění poštovních schránek. Podle toho, nakolik váš systém respektuje FHS, mohou být schránky stále umístěny ve /var/spool/mail.

/var/run

Adresář, do něhož se ukládají soubory obsahující informace o systému, jež platí až do jeho dalšího zavedení. Tak například soubor /var/run/utmp obsahuje informace o momentálně přihlášených uživatelích systému.

/var/spool

Adresáře pro elektronickou poštu, systém news, tiskové fronty a další subsystémy, které vyu-žívají metodu spoolingu a princip řazení úloh do fronty. Každý z těchto subsystémů má v tomto adresáři svůj vlastní podadresář, například news se ukládají do /var/spool/news. Pokud systém není plně kompatibilní s FHS, může ukládat poštovní schránky ve /var/spool/mail.

/var/tmp Do adresáře /var/tmp se ukládají velké dočasné soubory a dočasné soubory, které budou exi-stovat déle než ty, které se ukládají do adresáře /tmp. (Avšak správce systému by měl dbát na to, aby stejně jako v adresáři /tmp ani v adresáři /var/tmp nebyly uloženy velmi staré dočas-né soubory.)

Souborový systém /proc

Systém souborů /proc je vlastně imaginárním souborovým systémem. Ve skutečnosti na disku neexistuje. Místo toho jej v paměti

vytváří jádro systému. Ze systému souborů /proc lze získávat různé aktuální informace o systému (původně o procesech – z toho je odvozeno jeho jméno). Některé z významnějších souborů a adresářů popisujeme níže. Samotný souborový systém /proc je podrobněji popsán na manuálové stránce *proc*.

/proc/1

Adresář s informacemi o procesu číslo 1. Každý z procesů má v adresáři /proc vlastní podadresář, jehož jméno je stejné jako identifikační číslo procesu.

/proc/cpuinfo

Různé informace o procesoru. Například typ, výrobce, model, výkon a podobně.

/proc/devices

Seznam ovladačů zařízení konfigurovaných pro aktuálně běžící jádro systému.

/proc/dma

Informuje o tom, které kanály DMA jsou právě využívány.

/proc/filesystems

Souborové systémy konfigurované v jádru systému.

/proc/interrupts

Informuje o tom, která přerušení jsou využívána a kolikrát nastala.

/proc/ioports

Informuje o tom, které ze vstupně-výstupních portů se momentálně využívají.

/proc/kcore

Obraz fyzické paměti systému. Má velikost odpovídající velikosti fyzické paměti systému. Ve skutečnosti ale samozřejmě nezabírá takovéto množství paměti, protože jde o soubor genero-vaný „na požádání“, tedy pokaždé jenom v okamžiku, kdy k němu různé programy přistupu-jí. Uvědomte si, že soubory souborového systému /proc nezabírají ve skutečnosti (než je zko-pírujete na nějaké jiné místo na disku) vůbec žádný diskový prostor.

/proc/kmsg

Výstupní hlášení jádra systému. Zde uložená hlášení dostává i program syslog.

/proc/ksyms

Tabulka symbolů jádra systému.

/proc/loadavg

Statistika zatížení systému – tři celkem nic neříkající indikátory toho, kolik práce systém momentálně má.

/proc/meminfo

Informace o využití paměti, jak fyzické, tak virtuální.

/proc/modules

Informuje o tom, které moduly jádra jsou právě zavedeny v paměti.

/proc/net

Informace o stavu síťových protokolů.

/proc/self

Symbolický odkaz do adresáře procesů toho programu, který zrovna přistupuje k souborové-mu systému /proc. Když k systému souborů /proc současně přistupují dva různé procesy, budou mít přidělené dva různé odkazy. Tímto způsobem se mohou programy pohodlně a jed-noduše dostat k vlastnímu adresáři.

/proc/stat

Různé statistiky týkající se systému. Například počet výpadků stránek od zavedení systému a podobně.

/proc/uptime

Informuje o tom, jak dlouho systém běží.

/proc/version

Verze jádra systému.

Všechny výše uvedené soubory jsou normálně čitelné textové soubory, ne vždy jsou ale formáto-vány příliš přehledně. Existuje celá řada programů, které nedělají prakticky nic jiného, než že na-čtou některý z těchto souborů a zobrazí jej v přehlednějším formátu. Například program free pře-čte soubor /proc/meminfo a v něm uvedené údaje převede z bajtů na kilobajty (a ještě něco málo přidá). Zajímavé informace o systému /proc v češtině najdete například na adrese *http://www.root.cz/serialy/co-pred-nami-taji-proc/*.

V novějších verzích jádra 2.6 jsou některé informace dostupné také v adresáři /sys, což má na starost další virtuální souborový systém – sysfs. Ten byl navržen zejména s ohledem na potřebu získávat strukturované informace o hardwaru a využívá jej například již jednou zmíněný udev. Více informací najdete například na *http://en.wikipedia.org/wiki/Sysfs.*

Hardware, zařízení a nástroje

Znalost hovoří, moudrost naslouchá
– Jimi Hendrix

V této kapitole popisujeme, co jsou to soubory zařízení a jak se vytvářejí. Kanonický seznam sou-borů zařízení najdete v souboru /usr/src/linux/Documentation/devices.txt v případě, že máte nainstalovány zdrojové kódy jádra. Dále uvedené informace platí pro jádro 2.6.8.

Hardwarové nástroje

Skript MAKEDEV

Po instalaci operačního systému bude většina souborů zařízení již vytvořena a budou připraveny k použití. Pokud byste náhodou potřebovali nějaký, který neexistuje, vyzkoušejte nejprve skript MAKEDEV. Obvykle jej najdete jako /dev/MAKEDEV, případná kopie (nebo odkaz) může být iv /sbin/MAKEDEV. Pokud se nenachází ve spouštěcí cestě, budete muset cestu definovat explicitně.

Typicky se tento skript používá následujícím způsobem:

```
# /dev/MAKEDEV -v ttyS0 create ttyS0 c 4 64 root:dialout 0660
```

Tímto příkazem vytvoříte znakové zařízení /dev/ttyS0 s hlavním číslem 4 a vedlejším číslem 64,
jehož vlastníkem bude root, skupina dialout a práva budou 0660.Konkrétně ttyS0 je sériový port. Podle hlavního a vedlejšího čísla zařízení se orientuje jádro.Jádro přistupuje k zařízením výhradně pomocí čísel, což by znesnadňovalo orientaci normálníchsmrtelníků – proto používáme soubory s určitými názvy. Práva 0660 znamenají právo pro čtenía zápis vlastníkovi (root) a skupině (dialout). Ostatní uživatelé žádná práva k souboru nemají.

Příkaz mknod

Jako preferovaný způsob vytváření neexistujících zařízení se doporučuje skript MAKEDEV. V některých případech ale tento skript nemusí znát zařízení, které chcete vytvořit. Pak použijete příkaz mknod. Abyste jej mohli použít, musíte znát hlavní a vedlejší číslo zařízení, které chcete vytvořit. Základní zdroj těchto informací představuje soubor devices.txt v dokumentaci zdrojových kódů jádra. Předpokládejme pro ilustraci, že naše verze skriptu MAKEDEV neumí vytvořit zařízení /dev/ttyS0, proto je budeme vytvářet příkazem mknod. Z dokumentace k jádru se dozvíme, že má mít hlavní číslo 4 a vedlejší číslo 64. Teď už tedy víme všechno, abychom mohli zařízení vytvo-řit:

```
# mknod /dev/ttyS0 c 4 64 # chown root.dialout /dev/ttyS0 # chmod 0644 /dev/ttyS0 # ls -l /dev/ttyS0 crw-rw—— 1 root dialout 4, 64 Oct 23 18:23 /dev/ttyS0
```

Jak vidíte, v tomto případě je vytvoření zařízení o dost složitější. Ukázali jsme si nicméně celý potřebný postup. Je velmi nepravděpodobné, že by se soubor ttyS0 nevytvořil skriptem MAKEDEV, avšak pro ilustraci to stačí.

Další hardwarové prostředky

Další informace o tom, jaké hardwarové prostředky používá jádro, lze nalézt v adresáři /proc nebo /sys, viz kapitolu „Souborový systém /proc“.

Disky a jiná média

Na prázdném disku lze hledat věčně.

Při instalaci a aktualizaci systému vás u disků čeká hodně práce. Je potřeba vytvořit souborové systémy, do kterých se budou ukládat soubory, a vyhradit na discích prostor pro různé části systému.

Tato kapitola popisuje všechny tyto úvodní činnosti. Když tuto práci jednou podstoupíte a systém nastavíte, obvykle to už nebudete muset dělat znovu. Výjimkou je používání disket. K této kapi-tole se také budete vracet pokaždé, když budete přidávat nový pevný disk nebo pokud budete chtít optimálně vyladit diskový subsystém.

Mezi základní úkoly při správě disků patří:

Formátování pevného disku. Formátování disku je posloupností několika různých dílčích činností (jako je například kontrola výskytu vadných sektorů), které tento disk připravují na další použití. (V současnosti je většina nových pevných disků formátovaná výrobcem, a jejich formátování není nutné)
Rozdělení pevného disku na oblasti. Když chcete disk využívat pro několik činností, o kte-rých se nepředpokládá, že by se vzájemně ovlivňovaly, můžete jej rozdělit na samostatné diskové oblasti. Jedním z důvodů pro rozdělení disku na oblasti je instalace a provozová-ní různých operačních systémů na jednom disku. Jinou výhodou rozdělení pevného disku na oblasti je oddělení uživatelských souborů od souborů systémových. Tím se zjednoduší zálohování a sníží se pravděpodobnost poškození systémových souborů
Vytvoření souborového systému (vhodného typu) na každém disku nebo diskové oblasti. Z pohledu operačního systému nestačí disk pouze zapojit, Linux jej může používat až poté, co na něm vytvoříte nějaký souborový systém. Pak lze na disk ukládat soubory a přistu-povat k nim
Vytvoření jediné stromové struktury připojením různých souborových systémů. Systémy souborů se připojují buď automaticky nebo manuálně – podle potřeby. (Ručně připojené souborové systémy se obvykle musí také ručně odpojit)

Kapitola „Správa paměti“ obsahuje informace o virtuální paměti a diskové vyrovnávací paměti. Pra-cujete-li s disky, měli byste být s jejich principy obeznámeni

Dva druhy zařízení

Unix, a tedy i Linux znají dva různé typy zařízení. Jednak bloková zařízení s přímým přístupem (například disky) a jednak znaková zařízení (například pásky a sériové linky), která mohou být buď sériová nebo s přímým přístupem. Každému z podporovaných zařízení odpovídá v systému souborů jeden soubor zařízení. Když se čtou či zapisují data z anebo do souboru zařízení, přená-ší se ve skutečnosti na zařízení, které tento soubor reprezentuje. Takže pro přístup k zařízením nejsou nutné žádné zvláštní programy a žádné zvláštní programovací techniky (jako například ob-sluha přerušení nebo cyklické dotazování sériového portu). Když například chcete vytisknout nějaký soubor na tiskárně, stačí zadat:

```
$ cat jméno_souboru > /dev/lp1 $
```

a obsah souboru se vytiskne (soubor musí mít přirozeně nějakou formu, kterou tiskárna zná). Avšak vzhledem k tomu, že není příliš rozumné, aby několik uživatelů současně kopírovalo sou-bory na jedinou tiskárnu, pro tiskové úlohy se běžně používá zvláštní program (nejčastěji lpr). Tento program zajistí, že se v určitém okamžiku bude tisknout pouze jeden soubor. Ihned po ukončení jedné tiskové úlohy automaticky pošle na tiskárnu další úlohu. Podobný mechanismus přístupu vyžaduje většina zařízení. Takže ve skutečnosti se běžný uživatel o soubory zařízení skoro vůbec nemusí zajímat.

Protože se zařízení chovají jako soubory souborového systému (uložené v adresáři /dev), lze velmi lehce zjistit, které soubory zařízení existují. Lze použít například příkaz ls nebo jiný vhod-ný program. V prvním sloupci výstupu příkazu ls -l je uveden typ souboru a jeho přístupová práva.

Chcete-li si prohlédnout sériové zařízení systému, zadáte:

```
$ ls -l /dev/ttyS0 crw-rw-r— 1 root dialout 4, 64 Aug 19 18:56 /dev/ttyS0 $
```

Podle prvního písmene prvního sloupce, tedy písmene „c“ v řetězci „crw-rw-r—“, informovaný uživatel pozná, o jaký typ souboru jde. V tomto případě se jedná o znakové zařízení. U běžných souborů je prvním písmenem „-“, u adresářů je to „d“ a pro bloková zařízení se používá písmeno „b“. Podrobnější informace uvádí manuálová stránka k příkazu ls.

Všimněte si, že všechny speciální soubory obvykle existují, i když zařízení samotné není nainsta-lované. Takže například pouhý fakt, že v systému souborů je soubor /dev/sda, neznamená, že skutečně máte pevný disk SCSI. Díky tomu, že všechny speciální

soubory po instalaci operačního systému existují, lze zjednodušit instalační programy. Méně složité je tím pádem i přidávání nových hardwarových komponent (pro nově přidávané součásti již není třeba hledat správné para-metry a vytvářet speciální soubory).

Pevné disky

Tato kapitola zavádí terminologii, která se v oblasti pevných disků používá. Znáte-li tyto pojmy

a principy, můžete ji přeskočit.Na obrázku 5.1 jsou schematicky znázorněny důležité části pevného disku. Disk se skládá z jednénebo více kruhových *desek,* jejichž *povrchy* jsou pokryty magnetickou vrstvou, na kterou se zazna-menávají data. Každý povrch má svou *čtecí a zapisovací hlavu*, která čte nebo zaznamenává údaje.Desky rotují na společné hřídeli, typická rychlost otáčení je 5 400 nebo 7 200 otáček za minutu.(Pevné disky s vysokým výkonem používají i vyšší rychlosti.) Hlavy se pohybují podél rovinypovrchu, spojením tohoto pohybu s rotací povrchu může hlava přistupovat ke všem částem mag-netických povrchů. Procesor a disk spolu komunikují prostřednictvím *řadiče disku*. Díky řadiči se zbytek systému nemusí zajímat o to, jak pracovat s diskem, protože lze použít pro různé typy disků řadiče, jež mají pro ostatní součásti počítače stejné rozhraní. Takže počítač může disku (místo zadávání sérií dlouhých a složitých elektrických signálů, podle nichž se hlava nejdřív přesune na odpovídající místo disku, pak čeká, až se pod ni dostane žádaná pozice, a dělá další nepříjemnosti, které jsou pro danou operaci potřeba) jednoduše vzkázat: „Hele, milý disku, dej mně to, co potřebuji.“ (Ve skutečnosti je sice rozhraní řadiče stejně dost složité, ale rozhodně ne tak složité, jako by bylo v případě, kdyby ostatní prvky systému přistupovaly k disku přímo.) Řadič navíc může dělat některé další operace, obsluhuje například vyrovnávací paměť, automaticky nahrazuje vadné sek-tory a podobně.

Výše uvedené obvykle každému postačí k pochopení toho, jak hardware pevného disku funguje. Existuje pochopitelně ještě řada dalších částí, například motory, které otáčí diskem a přemisťují hlavy, elektronika, jež řídí operace dalších mechanických částí, a tak dále, jež ale většinou nejsou pro pochopení principu práce pevného disku tak důležité.

Magnetické vrstvy disku jsou obvykle rozděleny do soustředných kružnic, kterým se říká *stopy*. Stopy se dělí na *sektory*. Toto rozdělení se používá pro určování místa na disku a přidělování dis-kového prostoru souborům. Když například chcete na disku najít určité místo, zadáte „souřadni-ce“: vrstva 3, stopa 5, sektor 7. Počet sektorů je většinou pro všechny stopy stejný, ale některé pevné disky mají na vnějších stopách víc sektorů (všechny sektory mají stejnou fyzickou velikost, takže většina z nich je umístěna na delších, vnějších stopách). Typicky bude v jednom sektoru ulo-ženo 512 bajtů dat. Disk samotný pak neumí pracovat s menším množstvím dat, než je jeden sek-tor. Každý povrch je rozdělen na stopy a sektory stejným způsobem. Takže když je hlava jedno-ho povrchu nad určitou stopou, hlavy ostatních povrchů se také nachází nad odpovídajícími sto-pami. Všem těmto stopám dohromady se říká *cylindr*. Přemístění hlavy z jedné stopy (cylindru) na jinou trvá jistou dobu. Takže ukládáním dat, ke kterým se často přistupuje najednou (například data jednoho souboru), na stejný cylindr, se zamezí zbytečnému přesouvání hlav při jejich později-ším čtení. Snižuje se přístupová doba a zvyšuje výkon. Není ale vždycky možné uložit data na disk tímto způsobem. Souborům, které jsou na disku uloženy na několika místech, se říká *fragmento-vané*.

Počet povrchů (respektive hlav, což je to samé), cylindrů a sektorů se dost liší. Specifikace jejich počtu se nazývá *geometrie* pevného disku. Geometrie je obvykle uložena ve zvláštní bateriemi zálohované oblasti paměti, které se říká *CMOS RAM*, odkud si ji operační systém načítá vždy během zavádění nebo inicializace ovladače disku.

Naneštěstí má BIOS omezení, které neumožňuje zadat v paměti CMOS počet stop větší než 1 024, což je pro pevné disky velké kapacity příliš málo. Uvedené omezení lze obejít tak, že řadič pev-ného disku bude lhát ohledně skutečné geometrie disku a bude *překládat adresy* požadované systémem na adresy, které odpovídají realitě. Představme si například pevný disk, jenž má 8 hlav, 2 048 stop a 35 sektorů na stopu. Řadič tohoto disku bude zbytku systému lhát a tvrdit, že má 16 hlav, 1 024 stop a 35 sektorů na stopu, což nepřekračuje omezení v počtu stop. Pak bude při kaž-dém požadavku systému na přístup k disku překládat adresu, kterou dostane tak, že počet hlav vydělí dvěma a počet stop dvěma vynásobí. Matematické úpravy budou v praxi složitější, proto-že skutečná čísla nejsou tak „pěkná“ jako v uvedeném příkladu. Ale nezbývá než zopakovat, že detaily nejsou tak důležité pro pochopení principu. Překládání adres zkresluje pohled operačního systému na to, jak je disk ve skutečnosti organizovaný. To je nepraktické, protože nelze pro sní-žení přístupové doby a zvýšení výkonu použít „trik“ s ukládáním všech souvisejících dat na jeden cylindr.
Překlady adres jsou výlučně problémem disků typu IDE. Disky typu SCSI používají sekvenční čísla sektorů. To znamená, že řadič disku SCSI překládá každé sekvenční číslo sektoru na trojici [hlava, cylindr, sektor]. Navíc používá úplně jinou metodu komunikace s CPU, a proto jsou disky SCSI těchto problémů ušetřeny. Uvědomte si ale, že ani u disků SCSI počítač nezná jejich skutečnou geometrii.

Vzhledem k tomu, že operační systém Linux obvykle nezná skutečnou geometrii disku, nebudou se ani jeho souborové systémy pokoušet ukládat soubory na stejné cylindry. Místo toho se poku-sí souborům přidělit sekvenčně řazené sektory. Tato metoda téměř vždy zaručí podobný výkon, jakého lze dosáhnout při ukládání souvisejících dat na jeden cylindr. Celá problematika je o něco složitější, řadiče například využívají vlastní vyrovnávací paměť, nebo mechanismus automatické-ho, řadičem řízeného „přednačítání“ sekvenčně řazených sektorů.

Každý pevný disk je v systému reprezentován samostatným speciálním souborem. Typicky mohou být v systému maximálně dva

nebo čtyři pevné disky IDE. Zastupují je pak speciální soubory /dev/hda, /dev/hdb, /dev/hdc a /dev/hdd. Pevné disky SCSI reprezentují speciální soubory /dev/sda, /dev/sdb a tak dále. Podobné konvence týkající se názvů speciálních souborů platí i pro pevné disky jiných typů, podrobnosti najdete v kapitole „Hardware, zařízení a nástroje“. Pamatujte na to, že speciální soubory zastupující pevné disky umožňují přístup k disku jako celku, bez ohledu na diskové oblasti (o kterých budeme hovořit za chvíli). Není proto těžké práci s disky pochybit. Neopatrnost může v tomto případě vést ke ztrátě dat. Speciální soubory disků se obvy-kle používají pouze pro přístup k hlavnímu zaváděcímu sektoru disku, o kterém se také zmíníme později.

SAN (Storage Area Networks)

SAN je síť určená k ukládání dat s přístupem k LUN na úrovni bloků. LUN neboli číslo logické jed-notky (*logical unit number*) je virtuální disk v SAN. Systém má k LUN stejný přístup a stejná práva, jako kdyby to byl přímo připojený disk. Lze jej libovolně zformátovat a rozdělit na oblasti.

Obvykle se pro SAN používají dva síťové protokoly: *fibre channel* a *iSCSI*. Síť fibre channel je veli-ce rychlá a není zatěžovaná provozem firemní sítě LAN. Je však velice drahá. Jedna karta fibre channel stojí kolem 1 000 USD, dále jsou k provozu této sítě potřebné speciální přepínače.

iSCSI je novější technologie, jejímž prostřednictvím se posílají příkazy SCSI po síti TCP/IP. I když tato síť není tak rychlá jako síť fibre channel, hardware k ní je levnější.

NAS (Network Attached Storage)

NAS je metoda přístupu ke sdíleným diskům prostřednictvím stávající firemní sítě Ethernet na souborové úrovni. Disky není možné formátovat ani dělit na oblasti, neboť jsou sdílené několika počítači. Tato technologie se obvykle používá k tomu, aby několik pracovních stanic mělo přístup ke společným datům.

Podobně jako v případě SAN se pro přístup k diskům využívá protokol. V NAS je to buď
CIFS/Samba nebo NFS.
Protokol CIFS obvykle využívaly sítě Microsoft Windows, zatímco v sítích UNIX a Linux to byl pro
tokol NFS. Nyní mohou linuxové počítače prostřednictvím Samby také používat protokol CFIS.

Znamená to, že server Windows 2003 nebo linuxový počítač jsou servery NAS jenom proto, že zajišťují síťový přístup ke sdíleným diskům? Ano, jsou. Zařízení NAS si můžete koupit od celé řady výrobců. Tato zařízení jsou speciálně určena k zajištění vysokorychlostního přístupu k datům.

Diskety

Disketa sestává z pružné membrány pokryté z jedné nebo obou stran podobnou magnetickou sub-stancí jako pevný disk. Pružný disk samotný nemá čtecí a zápisovou hlavu, ta je součástí disketo-vé mechaniky. Disketa vlastně odpovídá jedné desce pevného disku, ale na rozdíl od pevného disku je vyměnitelná. Jednu disketovou mechaniku lze využít pro přístup k různým disketám, kdežto pevný disk je jedinou nedílnou jednotkou.

Podobně jako pevný disk se i disketa dělí na stopy a sektory. Dvě korespondující si stopy každé strany diskety tvoří cylindr. Počet stop a sektorů je ale pochopitelně o hodně nižší než u pevného disku.

Disketová jednotka umí obvykle pracovat s několika různými typy disket. Například 3,5palcová disketová mechanika může pracovat jak s disketami o velikosti 720 kB, tak 1,44 MB. Vzhledem k tomu, že disketová jednotka musí zacházet s každým typem diskety trochu jinak, musí i ope-rační systém vědět, který typ diskety je zrovna zasunutý v mechanice. Proto existuje pro jednotky pružných disků množství speciálních souborů, a to vždy jeden pro každou kombinaci typu dis-ketové jednotky a typu diskety. Takže soubor /dev/fd0H1440 pak reprezentuje první disketovou jednotku (fd0). Musí to být 3,5palcová mechanika pracující s 3,5palcovými disketami vysoké hus-toty záznamu (proto H v názvu zařízení) s kapacitou 1 440 kB (proto 1440). To jsou běžné 3,5pal-cové diskety HD.

Konstrukce tvorby názvů speciálních souborů zastupujících disketové jednotky je poměrně složi-tá. Proto má Linux pro diskety i zvláštní typy zařízení. Tato zařízení automaticky detekují typ dis-kety zasunuté v mechanice. Fungují tak, že se při požadavku na přístup pokouší přečíst první sek-tor vložené diskety, přičemž postupně zkouší jejich různé typy, až se jim podaří najít ten správný. Samozřejmě, podmínkou je, aby vložená disketa byla nejdříve naformátovaná. Automatická zaří-zení zastupují speciální soubory /dev/fd0, /dev/fd1 a tak dále.

Parametry, které tato automatická zařízení používají pro přístup k disketám, lze nastavit také pro-gramem setfdprm. To se může hodit jednak v případě, že používáte diskety, jež nemají běžnou kapacitu, to znamená, že mají neobvyklý počet sektorů, dále v případě, že autodetekce z nezná-mých důvodů selže anebo když vlastní speciální soubor chybí.

Kromě toho, že Linux zná všechny standardní formáty disket, umí pracovat i s množstvím nestan-dardních formátů. Některé z nich ale vyžadují speciální formátovací programy. Touto problemati-kou se teď nebudeme zabývat a doporučíme projít si soubor

/etc/fdprm. Ten blíže specifikuje nastavení, která program setfdprm podporuje.

Operační systém musí vědět o tom, že byla disketa v mechanice vyměněna. Jinak by totiž mohl například použít data z dříve vložené diskety, uložená ve vyrovnávací paměti. Bohužel, vodič, jenž se používá k signalizaci výměny diskety, bývá někdy poškozený. Při používání disketové jednot-ky v systému MS-DOS nebude mechanika vůbec schopná indikovat systému výměnu média, což je ještě horší situace. Jestli jste se někdy setkali s podivnými, zdánlivě nevysvětlitelnými problémy při práci s disketami, jejich příčinou mohla být právě nefunkční indikace výměny média. Jediným způsobem, jak lze tyto problémy odstranit, je nechat disketovou mechaniku opravit.

Jednotky CD-ROM

Jednotky CD-ROM čtou opticky data z plastických disků. Informace jsou zaznamenány na povr-chu těchto disků jako miniaturní „dolíčky“ seřazené v husté spirále, jež začíná uprostřed disku a končí na jeho okraji. Jednotka vysílá laserový paprsek, který čte data z disku tak, že sleduje tuto
spirálu. Od hladkého povrchu disku se odráží jinak, než když narazí na dolík. Tímto způsobem

lze jednoduše kódovat binární informace. Ostatní je prostě pouhá mechanika. Ve srovnání s pevnými disky jsou jednotky CD-ROM pomalé. Pevné disky mají průměrnou pří-stupovou dobu typicky menší než 15 milisekund, kdežto rychlá jednotka CD-ROM bude datavyhledávat s přístupovou dobou řádově v desetinách sekundy. Skutečná rychlost přenosu dat jeale celkem vysoká, zhruba stovky kilobajtů za sekundu. To, že je jednotka CD-ROM „pomalá“,znamená, že ji nelze pohodlně využít jako alternativu pevných disků, i když to samozřejmě možnéje. Některé distribuce Linuxu totiž nabízejí takzvané „live“ souborové systémy na CD-ROM. Tyfungují tak, že část souborů se při instalaci nakopíruje na pevný disk, a v případě potřeby se čtoupřímo z kompaktního disku. Instalace je pak jednodušší, méně časově náročná a ušetří se takéznačná část diskového prostoru. Jednotky CD-ROM lze s výhodou využít při instalaci nového pro-gramového vybavení, protože při instalování není vysoká rychlost určujícím faktorem.

Je několik způsobů, jak se data na disky CD-ROM ukládají. Nejrozšířenější z nich upravuje mezi-národní standard ISO 9660. Tato norma definuje minimální souborový systém, který je ještě o něcoprimitivnější než systém souborů používaný systémem MS-DOS. Na druhou stranu je souborovýsystém ISO 9660 tak jednoduchý, že by jej měly bez potíží dokázat mapovat na svůj vlastní sou-borový systém všechny operační systémy.

Pro běžnou práci v Unixu je ale souborový systém ISO 9660 prakticky nepoužitelný. Proto sezavedlo rozšíření tohoto standardu, kterému se říká „Rock Ridge“. Rock Ridge například umožňu-je používat dlouhá jména souborů, symbolické odkazy a mnoho dalších vymožeností, díky kte-rým se disk CD-ROM tváří víceméně jako unixový souborový systém. Navíc, rozšíření Rock Ridgezachovává kompatibilitu se souborovým systémem ISO 9660, takže je použitelné i v jiných nežunixových systémech. Operační systém Linux podporuje jak ISO 9660, tak Rock Ridge. Rozšířenírozezná a dále používá zcela automaticky.

Souborový systém je ale pouze jednou stranou mince. Většina disků CD-ROM často obsahuje data,ke kterým lze přistupovat výhradně pomocí zvláštních programů. Bohužel, většina těchto progra-mů není určena pro Linux (možná s výjimkou některých programů, jež běží pod linuxovým emu-látorem systému MS-DOS nebo pod wine, emulátorem Windows.

Kdybychom chtěli být ironičtí, řekli bychom, že wine nejspíš znamená „Wine Is Not an Emulator“(wine není emulátor). Wine je přesně řečeno náhradou API (Application Program Interface). Dalšíinformace naleznete na adrese *http://www.winehq.com.*

Existuje také komerční program VMWare, který softwarově emuluje celý virtuální počítač. Podrob

nosti viz adresa *http://www.vmware.com.*Mechanika CD-ROM je přístupná prostřednictvím odpovídajícího souboru zařízení. Mechanika CD-ROM může být připojena k počítači několika způsoby: pomocí SCSI (/dev/scd*, /dev/sr*),EIDE (/dev/hd*) nebo SATA (/dev/sd*) rozhraní, případně na velmi starých počítačích pomocízvukové karty. Popis hardwarových podrobností jde nad rámec této knihy, avšak způsobem při-pojení je určeno, jaký bude soubor zařízení.

Pásky

Magnetopáskové jednotky používají pásky podobné těm, které se používají pro záznam zvuku na magnetofonových kazetách. Páska je již svou povahou sériovým zařízením – aby bylo možné dostat se k některé její části, je nejdřív nutné projít všechny předchozí záznamy. K údajům na disku lze přistupovat přímo, takže je možné „skočit“ přímo na kterékoliv místo na něm. Sériový přístup k datům na páskách je příčinou toho, že jsou pomalé.
Na druhou stranu jsou pásky relativně levné, což kompenzuje nedostatky v rychlosti. Bez problé-mu je lze vyrobit dost dlouhé, takže je na ně možno uložit velké množství dat. To vše předurču-je pásková zařízení k archivaci a zálohování, u nichž se nevyžaduje vysoká rychlost, ale naopak, využívá se nízkých nákladů a velké kapacity.

Formátování

Formátování je procedura zápisu značek, které se používají na označení stop a sektorů na mag-netickém médiu. Předtím než je disk naformátován, jsou magnetické signály na jeho povrchu neu-spořádané, chaotické. Formátování do tohoto chaosu vnáší určitý řád. Načrtnou se – obrazně řeče-no – linie, ve kterých vedou stopy, a ty se pak rozdělí na sektory. Skutečné podrobnosti jsou tro-chu jiné, ale to není pro tuto chvíli podstatné. Důležité je to, že disk nelze používat bez toho, že by byl naformátován.

Pokud jde o formátování, je terminologie mírně zavádějící. V systémech MS-DOS a Windows se pod pojmem „formátování" rozumí i proces vytvoření souborového systému (o němž bude řeč později). Takže se obě tyto procedury označují jediným pojmem – obzvlášť u disket. Když je nutné je rozlišit, používá se pro formátování v pravém slova smyslu termín *nízkoúrovňové formátování* a pro vytvoření souborového systému označení *vysokoúrovňové formátování*. Terminologie pou-žívaná v unixovém světě označuje obě tyto činnosti tradičními pojmy „formátování" a „vytvoření souborového systému". Nejinak tomu bude i v této knize.

Disky IDE a některé disky SCSI jsou formátovány ve výrobě a není třeba je po připojení formáto-vat znovu, takže většina lidí se o formátování disků nestará. Vlastní naformátování disku dokon-ce může disku uškodit, protože některé disky vyžadují zvláštní způsob formátování, který pak umožňuje například automatické přemisťování vadných sektorů.

Disky, které je potřeba formátovat (a jež lze formátovat), vyžadují obvykle speciální formátovací programy, protože rozhraní formátovací logiky zabudované v jednotce se liší od jednoho typu disku k druhému. Takovéto formátovací programy jsou často buď součástí řadiče BIOS nebo běží pouze pod systémem MS-DOS. Ani jeden z těchto prostředků tedy nelze bez problémů použít v systému Linux.

V průběhu formátování můžete narazit na chybná místa na disku, kterým se říká *vadné bloky* nebo *vadné sektory*. S vadnými bloky si někdy poradí samotný řadič pevného disku, ale v případě, že se jich objeví víc, je potřeba nějakým opatřením zamezit možnému použití těchto vadných částí disku. Mechanismus, který problém vadných bloků řeší, je součástí souborového systému. Způ-sob, jakým systém souborů pracuje s informacemi o vadných sektorech, je popsán v dalších čás-tech této kapitoly. Jinou alternativou je vytvoření malé diskové oblasti, která by obsahovala pouze vadné bloky disku. Takovýto alternativní postup je vhodný zejména v případě, že je rozsah vad-ných sektorů velmi velký. Souborové systémy totiž mohou mít s rozsáhlými oblastmi vadných bloků problémy.

Diskety se formátují programem fdformat. Speciální soubor, který se má formátovat, se zadává jako jeho parametr. Následujícím příkazem bychom například zformátovali 3,5palcovou disketu s vysokou hustotou záznamu, vloženou do první disketové jednotky:

$ fdformat /dev/fd0H1440 Double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB. Formatting ... done Verifying ... done $

Pamatujte si, že když chcete použít zařízení s autodetekcí (například /dev/fd0), *musíte* nejdříve nastavit parametry zařízení pomocí programu setfdprm. Stejný výsledek jako v prvním příkladu mají příkazy:

$ setfdprm /dev/fd0 1440/1440 $ fdformat /dev/fd0 Double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB. Formatting ... done Verifying ... done $

Je obvykle výhodnější vybrat správný speciální soubor, jenž odpovídá typu diskety. Zapamatujte si, že není rozumné formátovat disketu na vyšší kapacitu, než je ta, pro kterou je určena. Program fdformat zároveň disketu prověří – zkontroluje, zda neobsahuje vadné bloky. Pokud narazí na vadný blok, opakovaně se ho pokusí použít (obvykle to uslyšíte – zvuky, které jednot-ka při formátování vydává, se dramaticky změní). Je-li na disketě pouze „dočasná" chyba (špatná úroveň signálu po zápisu znečištěnou zápisovou hlavou, planý poplach a podobně), program fdformat nic neřekne. Naopak skutečná chyba – fyzicky poškozený sektor – přeruší proces kon-troly povrchu diskety a jejího formátování. Jádro systému zapíše hlášení do logovacího souboru po každé vstupně-výstupní chybě, na kterou při formátování narazí. Tato hlášení se zároveň objeví na konzole. Když běží program syslog, zapisují se tato hlášení také do souboru /var/log/mes-sages. Samotným programem fdformat uživatel nezjistí, kde přesně se vadný blok nachází (obvy-kle to ani nikoho nezajímá – diskety jsou tak levné, že se ty vadné automaticky vyhazují).

$ fdformat /dev/fd0H1440 Double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB. Formatting ... done Verifying ... read: Unknown error $

Vadné bloky na disku nebo diskové oblasti (i disketě) lze vyhledat pomocí příkazu badblocks. Neprovádí formátování disku, takže je možné kontrolovat i disky, na nichž již existuje souborový systém. Níže uvedený příklad hledá chyby na 3,5palcové disketě se dvěma vadnými bloky:

```
$ badblocks /dev/fd0H1440 1440 718 719 $
```

Výstupem příkazu badblocks jsou čísla vadných sektorů. Většina souborových systémů umí tako-véto vadné bloky označit jako nepoužitelné. Systémy souborů udržují seznam, tabulku vadných sektorů, která se zakládá, když se systém souborů vytváří. Tento seznam pak lze kdykoliv měnit. První kontrola vadných bloků se dělá příkazem mkfs, jímž se vytváří souborový systém. Další kontroly se dělají programem badblocks a nové chybné bloky se přidávají do seznamu vadných bloků příkazem fsck. Příkazy

mkfs a fsck popíšeme později.

Moderní disky umí automaticky zjistit výskyt vadných bloků a pokouší se „opravit“ je tak, že místo nich použiji k těmto účelům zvlášť vyhrazené správné sektory. Mechanismus náhrady chybného bloku správným je pro operační systém transparentní. Pokud se zajímáte o podrobnosti, měly by být popsány v návodu k disku. Avšak i disky tohoto typu by teoreticky mohly selhat, kdyby počet vadných bloků výrazně vzrostl, nicméně s největší pravděpodobností disk „umře“ z jiných důvodů daleko dříve.

Diskové oblasti

Pevný disk může být rozdělen na několik *diskových oblastí*. Každá disková oblast se chová tak, jako by byla samostatným diskem. Diskové oblasti mají smysl v případě, že máte jeden pevný disk a chcete na něm používat například dva operační systémy – pak disk rozdělíte na dvě diskové oblasti a každý operační systém bude používat vlastní diskovou oblast a nebude zasahovat do druhé. Takto mohou oba operační systémy v klidu a míru koexistovat na jediném disku. Kdybys-te nepoužili rozdělení disku na samostatné diskové oblasti, museli byste zakoupit pevný disk pro každý z operačních systémů.

Diskety se na diskové oblasti nerozdělují. Z technického hlediska to možné je, ale vzhledem k tomu, že mají malou kapacitu, by oblasti byly prakticky využitelné jenom v ojedinělých přípa-dech. Ani disky CD-ROM se obvykle nedělí na oblasti, protože je praktičtější je používat jako jeden velký disk a skutečně málokdy je potřeba mít na jednom disku CD-ROM uloženo několik operačních systémů.

Hlavní zaváděcí sektor, zaváděcí sektory a tabulka diskových oblastí

Informace o tom, jak je pevný disk rozdělen na diskové oblasti, je uložena v prvním sektoru (tj. prvním sektoru první stopy první vrstvy disku). Tento sektor obsahuje takzvaný *hlavní zaváděcí záznam* (master boot record, MBR). Je to sektor, který se načítá a spouští systémem BIOS pokaž-dé, když se počítač spustí. Zaváděcí sektor disku obsahuje krátký program, jenž načte tabulku diskových oblastí a zjistí, která oblast disku je aktivní (tedy označená jako zaváděcí). Pak přečte první sektor této oblasti, takzvaný *zaváděcí sektor*. (MBR je také zaváděcí sektor, ale má zvláštní posta-vení, a proto i zvláštní označení.) Zaváděcí sektor na diskové oblasti obsahuje další krátký pro-gram, který načítá první část operačního systému, jenž je na této diskové oblasti uložený (samozřejmě za předpokladu, že je disková oblast bootovatelná), a spouští jej.

Metoda dělení disku na diskové oblasti není hardwarově implementovaná a není ani součástí systému BIOS. Jde čistě o konvenci podporovanou většinou operačních systémů. Ale ne všechny operační systémy se chovají podle těchto konvencí, jsou i výjimky. Některé operační systémy sice podporují dělení disku na diskové oblasti, ale v rámci své diskové oblasti používají svou vlastní interní metodu dělení. Tyto typy operačních systémů mohou bez jakýchkoliv zvláštních prostřed-ků spolupracovat s jinými systémy (včetně operačního systému Linux). Naopak, takový operační systém, jenž rozdělení disku na diskové oblasti nepodporuje, nemůže koexistovat na stejném pev-ném disku s jiným operačním systémem.

Je dobré si na kousek papíru vypsat tabulku rozdělení disku na oblasti. Kdyby pak náhodou v budoucnu došlo k poškození některé oblasti, nemusíte díky tomuto jednoduchému bezpeč-nostnímu opatření přijít o všechna data. (Poškozenou tabulku oblastí lze opravit programem fdisk.) Důležité informace získáte příkazem fdisk -l:

```
$ fdisk -l /dev/sda Disk /dev/sda: 100.0 GB, 100030242816 bytes 240 heads, 63 sectors/track, 12921 cylinders Units = cylindry of 15120 * 512 = 7741440 bytes

  Zařízení Boot Start End Blocks Id System /dev/sda1 * 1 934 7061008+ 83 Linux /dev/sda2 935 12921 90621720 5 Rozšířený /dev/sda5 935 1204 2041168+ 82 Linux swap /dev/sda6 1205 12921 88580488+ 83 Linux
```

Rozšířené a logické diskové oblasti

Původní schéma dělení disků počítačů PC na diskové oblasti umožňuje vytvořit pouze čtyři oblas-ti. To se záhy v praxi ukázalo jako nedostatečné. Zčásti například proto, že někteří uživatelé chtě-li mít na svém počítači i víc než čtyři operační systémy (Linux, MS-DOS, OS/2, Minix, FreeBSD, NetBSD nebo Windows/NT, a to jsme vyjmenovali jenom některé), ale především proto, že je výhodné mít několik diskových oblastí i pro jeden operační systém. Například z důvodů zlepšení odezvy operačního systému je lepší nevyužívat pro odkládací prostor systému Linux hlavní dis-kovou oblast operačního systému, ale mít odkládací prostor na samostatné diskové oblasti (viz dále).

Aby bylo možné omezení počtu diskových oblastí obejít, byly zavedeny takzvané *rozšířené dis-kové oblasti*. Tento trik umožňuje rozdělit *primární diskové oblasti* na podoblasti. Takto rozdělená primární oblast je onou rozšířenou oblastí a její části (podoblasti) jsou takzvané *logické diskové oblasti*. Chovají se stejně jako primární, ale jsou vytvořeny jiným způsobem. Není mezi nimi ale žádný rozdíl v rychlosti.

Struktura oblastí pevného disku by mohla vypadat například tak, jak je uvedeno na obrázku 5.2. Disk je rozdělen na tři primární diskové oblasti. Druhá z nich je rozdělena na dvě logické. Část disku nepatří vůbec žádné diskové oblasti. Disk jako celek a každá primární oblast má svůj zaváděcí sektor.

Typy diskových oblastí

Tabulky rozdělení disku (jedna v MBR a další na rozšířených diskových oblastech) mají vyhraze-ný jeden bajt pro každou diskovou oblast a ten identifikuje její typ. Snaží se popsat operační systém, který diskovou oblast používá, nebo její účel. Informační bajt by měl zamezit tomu, aby dva operační systémy pracovaly s jednou diskovou oblastí. Avšak ve skutečnosti většina operač-ních systémů označení typu diskové oblasti ignoruje. I Linux se například vůbec nezajímá o to, jak je určitá disková oblast označená. Dokonce – což je ještě horší – některé operační systémy tento bajt používají nesprávně. Například přinejmenším některé verze systému DR-DOS ignorují nejvyšší bit tohoto bajtu.

Žádný z úřadů pro normalizaci nespecifikoval, co která hodnota bajtu znamená. Některé obecně přijaté hodnoty a odpovídající typy uvádí tabulka 5.1. Obsáhlejší seznam vypíše linuxový program fdisk.

0	prázdný	1c	Hidden Win95 FA	70	DiskSecure Mult	bb	Boot Wizard hid
1	FAT12	1e	Hidden Win95 FA	75	PC/IX	be	Solaris boot
2	XENIX root	24	NEC DOS	80	Old Minix	c1	DRDOS/sec (FAT
3	XENIX usr	39	Plan 9	81	Minix / old Lin	c4	DRDOS/sec (FAT
4	FAT16 <32M	3c	PartitionMagic	82	Linux swap	c6	DRDOS/sec (FAT
5	Extended	40	Venix 80286	83	Linux	c7	Syrinx
6	FAT16	41	PPC PReP Boot	84	OS/2 hidden C:	da	Non-FS data
7	HPFS/NTFS	42	SFS	85	Linux extended	db	CP/M / CTOS / .
8	AIX	4d	QNX4.x	86	NTFS volume set	de	Dell Utility
9	AIX bootable	4e	QNX4.x 2nd part	87	NTFS volume set	df	BootIt
a	OS/2 Boot Manag	4f	QNX4.x 3rd part	8e	Linux LVM	e1	DOS access
b	Win95 FAT32	50	OnTrack DM	93	Amoeba	e3	DOS R/O

<zkráceno>

Typy diskových oblastí (podle programu fdisk)

Dělení pevného disku na diskové oblasti

Je mnoho programů, které umí vytvářet a mazat diskové oblasti. Součástí většiny operačních systé-mů je nějaký takový a bývá rozumné používat program, jenž je součástí operačního systému, v jehož prostředí s diskovými oblastmi pracujete, hlavně proto, kdyby dělal něco speciálního, co jiné nedělají. Většinou se tyto programy jmenují fdisk (včetně toho, který je součástí distribuce systému Linux) nebo nějak podobně. Detaily týkající se možností linuxového programu fdisk podrobně popisuje jeho manuálová stránka. Podobný je příkaz cfdisk, který má hezčí, celoobra-zovkové uživatelské rozhraní.

V případě, že používáte disky IDE, musí být celá zaváděcí disková oblast (tedy disková oblast, na které jsou uloženy soubory obrazů jádra) na prvních 1 024 cylindrech. To proto, že při zavádění operačního systému (předtím než systém přechází do chráněného režimu) se k disku přistupuje přes BIOS a ten neumí pracovat s více než 1 024 cylindry. V některých případech je možné pou-žívat zaváděcí diskovou oblast, která leží na prvních 1 024 cylindrech, jenom částečně. To je možné, jenom když na prvních 1 024 cylindrech budou uloženy všechny soubory, které při inici-alizaci čte systém BIOS. Vzhledem k tomu, že takovéhoto uspořádání je velmi obtížné dosáhnout, *je lepší je vůbec nepoužívat* – nikdy si totiž nemůžete být jistí, zda změna parametrů jádra systému nebo defragmentace disku nezpůsobí, že systém nebude vůbec možné zavést. Proto se raději vždy ujistěte, že je zaváděcí disková oblast vašeho systému celá na prvních 1 024 cylindrech.

Toto omezení již neplatí pro novější verze programu LILO přítomné v nových distribucích, které umí pracovat s LBA (Logical Block Addressing). V dokumentaci vaší distribuce byste měli zjistit, zda vaše verze LILO tento mechanismus podporuje.

Nové verze systémů BIOS a disků IDE již umí pracovat i s disky, které mají více než 1 024 cylin-drů. Máte-li takovýto systém, můžete na tento problém zapomenout. Jestli si v tom nejste zcela jisti, umístěte raději podle doporučení zaváděcí diskovou oblast na prvních 1 024 cylindrů.

Každá disková oblast by měla mít sudý počet sektorů, protože souborové systémy Linuxu použí-vají diskové bloky o velikosti 1 kB, tedy dva sektory. Lichý počet sektorů diskové oblasti způso-bí, že poslední sektor bude nevyužitý. Nemělo by to způsobit žádné zvláštní problémy, ale není to příliš elegantní. Některé verze programu fdisk vás budou na tento stav upozorňovat.

Při změně velikosti diskové oblasti je nejlepší vytvořit zálohu všeho, co chcete na diskovou oblast zachránit (a ještě lepší je zálohovat úplně všechno), pak diskovou oblast smazat, vytvořit novou a obnovit na ní soubory ze zálohy. Chcete-li zvětšit velikost nějaké diskové oblasti, budete muset pravděpodobně upravit i velikosti (tedy vytvořit zálohy a obnovit soubory) sousedních diskových oblastí.

Vzhledem k tomu, že změny velikostí diskových oblastí jsou dost pracné, je lepší nastavit je správ-ně hned napoprvé. Jinak budete potřebovat efektivní a jednoduchý systém zálohování. Instaluje-te-li poprvé systém z médií, jež nevyžadují časté zásahy obsluhy (například z CD-ROM – proti-kladem jsou diskety), je často jednodušší si nejdřív pohrát s různými konfiguracemi. Jelikož zatím na disku nemáte žádná data, která by bylo potřeba zálohovat, není natolik bolestné několikrát změnit velikosti diskových oblastí.

Existuje program pro systém MS-DOS, který se jmenuje fips, a ten umí změnit velikosti diskových oblastí systému MS-DOS bez toho, že by bylo potřeba zálohovat, mazat a obnovovat soubory z těchto diskových oblastí. Pro jiné souborové systémy je to zatím nadále nevyhnutelné.

Program fips je součástí většiny linuxových distribucí. Existuje i komerční program pro správu oblastí, „Partition Magic“, který má podobnou funkci s hezčím rozhraním. Je nutno si uvědomit, že přerozdělování disků na oblasti je činnost velmi nebezpečná. Bezpodmínečně se musíte přesvěd-čit, zda máte aktuální zálohy všech důležitých dat, než se pustíte do měnění oblastí „za chodu“. Velikosti diskových oblastí včetně oblastí pro MS-DOS umí změnit i program parted, avšak někte-ré jeho funkce jsou omezené. Nejdříve je ovšem nutné si prostudovat dokumentaci k tomuto pro-gramu; i zde „platí dvakrát měř a jednou řež“. Některé distribuce obsahují vlastní (zpravidla GUI) nástroje pro změnu velikosti oddílů – např. openSUSE, Mandriva Linux či Fedora Core.

Speciální soubory a diskové oblasti

Každá disková oblast a rozšířená disková oblast má svůj vlastní soubor zařízení. Podle konvence pro konstrukci názvů souborů zařízení se číslo diskové oblasti připojí za jméno celého disku s tím, že 1–4 budou primární diskové oblasti (podle toho, kolik primárních diskových oblastí bylo vytvo-řeno) a 5–8 jsou logické diskové oblasti (bez ohledu na to, na které z primárních diskových oblas-tí jsou vytvořeny). Například /dev/hda1 je první primární disková oblast prvního pevného disku IDE a /dev/sdb7 je třetí rozšířená disková oblast na druhém pevném disku SCSI.

Souborové systémy

Co jsou to souborové systémy?

Souborový systém tvoří metody a struktury dat, pomocí kterých operační systém udržuje záznamy o souborech na discích a diskových oblastech. Jde tedy o způsob, jakým jsou soubory na disku organizované. Tento termín se ale používá i k označení diskové oblasti nebo disku, na kterém se ukládají soubory, nebo ve významu typu souborového systému. Takže když někdo řekne, že „má dva souborové systémy“, může mít na mysli to, že má disk rozdělen na dvě diskové oblasti, nebo to může znamenat, že používá „souborový systém ext2“, tedy určitý typ souborového systému.

Rozdíl mezi diskem či diskovou oblastí a souborovým systémem, který je na nich vytvořený, je dost podstatný. Jen některé programy (mezi nimi – zcela logicky – programy, pomocí kterých se souborové systémy vytváří) pracují přímo se sektory disku nebo diskové oblasti. Jestli na nich byl předtím vytvořený systém souborů, bude po spuštění takovýchto programů zničen nebo vážně poškozen. Většina aplikací ale pracuje se souborovým systémem. Proto je nelze použít na disko-vé oblasti, na které není vytvořen žádný systém souborů (nebo na které je vytvořen souborový systém nesprávného typu).

Předtím než bude možné diskovou oblast nebo disk použít, je potřeba je inicializovat – musí se na ně zapsat určité datové struktury. Tento proces se označuje jako *vytvoření souborového systému*. Většina unixových souborových systémů má podobnou obecnou strukturu, v dalších podrobnos-tech se ale celkem dost liší. Mezi ústřední pojmy patří *superblok*, *inode*, *datový blok*, *adresářový blok* a *nepřímý blok*. Superblok obsahuje informace o souborovém systému jako celku, například jeho velikost (zrovna u této položky závisí přesné hodnoty na konkrétním souborovém systému). Inode obsahuje všechny informace o souboru kromě jeho jména. Jméno souboru je uloženo v adresáři společně s odpovídajícím číslem inode. Adresářová položka obsahuje jména souborů a čísla inodů, které tyto soubory reprezentují. Inode dále obsahuje čísla datových bloků, v nichž jsou uložená data souboru, který daný inode zastupuje. V inode je ale místo jenom pro několik čísel datových bloků. Když je jich potřeba víc, je dynamicky alokováno víc místa pro další uka-zatele na datové bloky. Tyto dynamicky alokované bloky jsou uloženy v nepřímém bloku. Jak jejich název naznačuje, v případě, že je potřeba najít datový blok, musí se nejdřív najít jeho číslo v nepřímém bloku.

Unixové souborové systémy obvykle umožňují vytvořit v souboru *díru*, a to pomocí systémové-ho volání lseek() (podrobnosti uvádí příslušná manuálová stránka). Vypadá to tak, že souborový systém předstírá, že je na konkrétním místě souboru uložen blok nulových bajtů, nicméně pro něj není vyhrazen žádný sektor pevného disku (to znamená, že takovýto soubor zabírá o něco méně diskového prostoru). S tím se můžete setkat zvlášť často u malých binárních souborů, sdílených knihoven Linuxu, některých

databází a v několika dalších speciálních případech. (Díry jsou imple-mentovány prostřednictvím speciální adresy datového bloku v nepřímém bloku nebo inode. Tato zvláštní adresa znamená, že pro určitou část souboru není alokován žádný datový blok, tedy že je v tomto souboru díra.)

Galerie souborových systémů

Linux podporuje několik typů souborových systémů. Mezi nejdůležitější patří:

minix

Nejstarší a pravděpodobně nejspolehlivější systém s omezenými možnostmi (chybějí časové známky, jména souborů jsou omezena na 30 znaků) a rozsahy (maximálně 64 MB na souborový systém).

xia

Modifikovaná verze souborového systému minix s rozšířenými limity jmen souborů a velikos-ti souborových systémů, avšak bez nových funkcí. Není příliš oblíbený, má však velmi dobrou pověst.

ext3

Souborový systém ext3 má všechny vlastnosti jako souborový systém ext2. Liší se v tom, že přibyl žurnál. Zvýšil se tím výkon a doba zotavení v případě havárie systému. Stal se oblíbenějším než ext2.

ext2

Předchůdce ext2, bez žurnálu. Nové verze jsou kompatibilní se staršími, takže není třeba měnit stávající souborové systémy.

ext

Starší verze ext2, která není kompatibilní s novými verzemi. V nových instalacích se už nejspíš nepoužívá a většina uživatelů přešla na ext2.

reiserfs

Velmi robustní souborový systém. Používá žurnál, čímž snižuje nebezpečí ztráty dat. Žurnál je mechanismus záznamu transakce, která se má provést nebo která se provedla. To umožňuje snadnou rekonstrukci souborového systému, který byl poškozen například nesprávným vypnutím systému.

jfs

JFS je žurnálovaný souborový systém navržený IBM pro vysoké výkony.

xfs

Souborový systém XFS původně navrhla firma Silicon Graphics jako 64bitový žurnálovaný systém. Byl určen pro zpracování velkých souborů na velkých souborových systémech.

Poznámka Některé souborové systémy používané i jinde mají natolik dobrou podporu, že lze měnit soubory mezi různými operačními systémy. Tyto systémy pracují stejně jako systémy původní, pouze mohou postrádat některé unixové vlastnosti, mohou mít určitá neočekávaná omezení nebo jiné zvláštnosti.

msdos

Kompatibilní se souborovými systémy MS-DOS (a OS/2 a Windows NT).

umsdos

Rozšiřuje souborový systém msdos o dlouhá jména souborů, vlastníky, přístupová práva a sou-bory zařízení. To umožňuje normálnímu souborovému systému msdos, aby byl používán jako linuxový souborový systém, takže není nutná existence samostatné oblasti pro Linux.

vfat

Rozšíření souborového systému FAT známé pod jménem FAT32. Podporuje větší velikosti disků než FAT. Většina disků MS Windows je vfat.

iso9660

Standardní souborový systém CD-ROM; oblíbené rozšíření Rock Ridge na standard CD-ROM, který umožňuje používání delších jmen souborů; je podporováno automaticky.

nfs

Síťový souborový systém, který umožňuje sdílení souborových systémů mezi více počítači. Důvodem je zajištění snazšího přístupu k souborům ze všech počítačů.

smbfs (cifs) Síťový souborový systém, který umožňuje sdílení souborových systémů s počítači vybavenými operačním systémem

MS Windows. Je kompatibilní s protokoly systému Windows pro sdílení souborů.

hpfs

Souborový systém operačního systému OS/2.

sysv

Souborové systémy operačních systémů SystemV/386, Coherent a Xenix.

NFTS

Nejnovější žurnálovaný souborový systém firmy Microsoft s rychlejším přístupem a stabilitou vyšší, než měly předchozí souborové systémy firmy Microsoft.

Výběr souborového systému závisí na situaci. Pokud vás kompatibilita či jiné důvody přinutí k výměně souborového systému, není vyhnutí. Může-li si člověk vybrat, pravděpodobně nejlepší je ext3, neboť má všechny vlastnosti systému ext2 a je žurnálovaný. Další informace o souboro-vých systémech najdete v kapitole „Porovnání souborových systémů“. Návod k souborovým systé-mům naleznete také na adrese *http://www.tldp.org/HOWTO/Filesystems-HOWTO.html.*

Dalším ze souborových systémů je proc, nejčastěji dostupný prostřednictvím adresáře /proc. Ve skutečnosti ale není souborovým systémem v pravém slova smyslu, i když tak na první pohled vypadá. Systém souborů proc umožňuje přístup k určitým datovým strukturám jádra systému, například k seznamu procesů (odtud jeho jméno). Přizpůsobuje tyto datové struktury tak, že se navenek chovají jako soubory souborového systému. K systému souborů proc pak lze přistupo-vat všemi běžnými nástroji, které se soubory běžně pracují. Takže kdybyste chtěli znát například

seznam všech procesů, zadali byste příkaz:

```
$ ls -l /proc -r—r—r— 1 root -r—r—r— 1 root -r—r—r— 1 root -r—r—r— 1 root -r—r—r— 1 root dr-xr-xr-x 2 root -r—r—r— 1 root -r—r—r
— 1 root -r—r—r— 1 root dr-xr-xr-x 18 root -r—r—r— 1 root -r————— 1 root -r—r—r— 1 root -r—r—r— 1 root -r—r—r— 1 root
lrwxrwxrwx 1 root -r—r—r— 1 root dr-xr-xr-x 9 root dr-xr-xr-x 4 root dr-xr-xr-x 4 root dr-xr-xr-x 4 root <zkráceno> root root root root root root
root root root root root root root root root root root root root root root root
20193 led 15 10:29 config.gz 0 led 15 06:59 cpuinfo 0 led 15 10:29 devices 0 led 15 10:29 dma 0 led 15 10:29 filesystems 0 led 15 10:29 ide/ 0
      led 15 10:29 interrupts 0 led 15 10:29 iomem 0 led 15 10:29 ioports 0 led 15 10:29 irq/ 0 led 15 10:29 kallsyms
939524096 led 15 10:29 kcore
                                                    0 led 15 10:29 loadavg
                                                    0 led 15 10:29 meminfo
    0 led 15 10:29 partitions
   64 led 15 06:53 self -> 21451/
                                                    0 led 15 10:29 swaps
                                                    0 pro 18 10:05 sys/
    0 led 12 15:59 1/
                                                    0 led 15 04:15 1048/
                                                    0 led 15 04:15 116/
```

(V seznamu bude vždy několik souborů, které neodpovídají žádným procesům. Výstup ve výše uvedeném příkladu byl zkrácen.) Je důležité si uvědomit, že i když se pro systém proc používá označení „souborový systém“, žádná z jeho částí neleží na žádném z disků. Existuje jenom „v představách“ jádra systému. Kdykoliv k některé části souborového systému proc přistupuje uživatel systému nebo některý z procesů, jádro „předstírá“, že je tato část uložená na disku, ale ve skutečnosti tomu tak není. Takže i když je v souborovém systému proc uložený poměrně veliký soubor /proc/kcore, ve skutečnosti nezabírá na disku žádné místo.

Který souborový systém použít?

Obvykle není moc důvodů používat několik různých souborových systémů. V současné době je nejoblíbenějším souborový systém ext3, neboť je žurnálovaný. Současně je to pravděpodobně nej-rozumnější volba. Ve vztahu ke zmiňovaným účetním strukturám, rychlosti, (pochopitelně) spo-lehlivosti, kompatibilitě a vzhledem k různým jiným důvodům by mohlo být vhodné zvolit i jiný systém souborů. Takovéto požadavky je třeba posuzovat případ od případu.

Souborový systém, který používá žurnál (záznam o prováděných operacích), se nazývá žurnálovaný. Vše, co se přihodilo v systému, se zapisuje do žurnálu. V případě havárie systému nebo když váš dvouletý synek stiskne síťový vypínač, jak to s oblibou dělá můj syn, slouží žurnál k obnově neuložených a ztracených dat. Ztráta dat je tak méně pravděpodobná. Žurnál se již stal výbavou linuxových souborových systémů, nepodléhejte však falešnému pocitu bezpečí. K chybě může dojít kdykoli a data by měla být pro případ nouze vždy zálohovaná.

Další podrobnosti o různých souborových systémech najdete v kapitole „Porovnání souborových systémů“.

Vytváření souborového systému

Souborové systémy se vytváří a inicializují příkazem mkfs. Je to vlastně vždy jiný program pro každý typ souborového systému. Program mkfs je jenom koncové rozhraní, program, který spouš-tí některé další programy, podle typu požadovaného souborového

systému. Typ souborového systému se volí přepínačem -t fstype.

Programy volané příkazem mkfs mají nepatrně odlišné rozhraní příkazové řádky. Běžně použí-vané a nejdůležitější volby jsou uvedeny níže. Více informací najdete v manuálových stránkách.

-t fstype

Typ souborového systému.

Vyhledat a ošetřit chybné bloky.

-l filename

Přečíst seznam vadných bloků ze souboru.

Kdybyste chtěli vytvořit souborový systém ext2 na disketě, zadali byste následující příkazy:

$ fdformat -n /dev/fd0H1440 Double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB. Formatting ... done $ badblocks /dev/fd0H1440 1440 > bad-blocks $ mkfs -t ext2 -l bad-blocks /dev/fd0H1440 mke2fs 0.5a, 5-Apr-94 for EXT2 FS 0.5, 94/03/10 360 inodes, 1440 blocks 72 blocks (5.00%) reserved for the super user First data block=1 Block size=1024 (log=0) Fragment size=1024 (log=0) 1 block group 8192 blocks per group, 8192 fragments per group 360 inodes per group Writing inode tables: done Writing superblocks and filesystem accounting information: done $

V prvním kroku se disketa formátuje (volba -n zakáže její validaci, tedy kontrolu vadných sekto-rů). Vadné bloky pak vyhledává program badblocks, a to s výstupem přesměrovaným do sou-boru bad-blocks. Nakonec se vytvoří souborový systém a mezi účetní struktury se uloží seznam vadných bloků, ve kterém budou všechny vadné sektory, jež našel program badblocks.

Místo příkazu badblocks je možno použít parametr -c programu mkfs tak, jak to uvádí následu-jící příklad:

$ mkfs -t ext2 -c /dev/fd0H1440 mke2fs 0.5a, 5-Apr-94 for EXT2 FS 0.5, 94/03/10 <zkráceno> Checking for bad blocks (read-only test): done Writing inode tables: done Writing superblocks and filesystem accounting information: done $

Volba -c programu mkfs je sice výhodnější než badblocks, ale program badblock je nutné pou-
žít pro kontrolu vadných bloků na již vytvořeném souborovém systému.Postup, kterým se vytváří souborové systémy na pevných discích a diskových oblastech, je stejnýjako naznačený postup inicializace souborového systému na disketě, s tím rozdílem, že není nutnéje formátovat.

Velikost bloku v souborovém systému

Velikost bloku udává velikost, kterou bude systém používat pro čtení a zápis dat. Větší bloky zvýšívýkon disku při práci s velkými soubory, například s databázemi. Důvodem je skutečnost, že diskse může déle věnovat čtení nebo zápisu, než začne hledat další blok.

Nevýhodou je, že máme-li v tomto souborovém systému mnoho menších souborů, např. v adre

sáři /etc, zůstane na disku hodně nevyužitého místa.Když například nastavíte velikost bloku na 4096, resp. 4k, a vytvoříte soubor o velikosti 256 bajtů,stejně spotřebuje 4k diskového prostoru. U jednoho souboru se nic neděje, když však souborovýsystém obsahuje stovky nebo tisíce souborů, nevyužitý prostor se nasčítá.

Velikost bloku má v některých systémech vliv na maximální velikost souboru. Důvodem je to, žemnohé moderní systémy nemají omezení, která se vztahují pouze na velikost bloku a velikost sou-boru, nýbrž i na počet bloků. Proto byste měli používat vztah „velikost bloku * max # bloků =max velikost bloku“.

Porovnání souborových systémů

Jméno Rok uvedení Původní OS Max. velikost Max. velikost Žurnál soub. systému souboru soub. systému

FAT16	1983	MSDOS V2	4 GB	16 MB až 8 GB	Ne
FAT32	1997	Windows 95	4 GB	8 GB až 2 TB	Ne
HPFS	1988	OS/2	4 GB	2 TB	Ne
NTFS	1993	Windows NT	16 EB	16 EB	Ano
HFS+	1998	Mac OS	8 EB	?	Ne
UFS2	2002	FreeBSD	512 GB až 32 PB	1 YB	Ne
ext2	1993	Linux	16 GB až 2 TB4	2 TB až 32 TB	Ne
ext3	1999	Linux	16 GB až 2 TB4	2 TB až 32 TB	Ano

Jméno soub. systému	Rok uvedení	Původní OS	Max. velikost souboru	Max. velikost soub. systému	Žurnál
ReiserFS3	2001	Linux	8 TB	16 TB	Ano
ReiserFS4	2005	Linux	?	?	Ano
XFS	1994	IRIX	9 EB	9 EB	Ano
JFS	?	AIX	8 EB	512 TB až 4 PB	Ano
VxFS	1991	SVR4.0	16 EB	?	Ano
ZFS	2004	Solaris 10	1 YB	16 EB	Ne

Porovnání vlastností souborových systémů

kilobajt - kB 1024 bajtů megabajt - MB 1024 kB gigabajt - GB 1024 MB terabajt - TB 1024 GB petabajt - PB 1024 TB exabajt - EB 1024 PB zetabajt - ZB 1024 EB jotabajt - YB 1024 ZB

Velikosti

Dlužno poznamenat, že s exabajty, zetabajty a jotabajty se setkáváme zřídka, pokud vůbec. Odhaduje se, že veškeré tištěné materiály na světě mají rozsah 5 exabajtů. Proto jsou některá omezení souborových systémů považována za víceméně teoretická. Souborové systémy však byly napsány pro tyto rozsahy.

Podrobnější informace naleznete na *http://en.wikipedia.org/wiki/Comparison_of_file_systems.*

Připojení a odpojení

Souborový systém se musí před použitím *připojit*. Po připojení systému souborů dělá operační systém některé účetní operace, kterými se ověřuje funkčnost připojení. Protože všechny soubory v Unixu jsou součástí jediného hierarchického adresářového stromu, operace připojení souboro-vého systému začlení obsah připojovaného souborového systému do některého z adresářů dříve připojeného systému souborů.

Na obrázku 5.3 jsou například zobrazeny tři samostatné souborové systémy, každý se svým vlast-ním kořenovým adresářem. Když budou poslední dva souborové systémy připojeny pod adresá-ře /home a /usr prvního systému souborů, dostaneme jeden adresářový strom, který je vyobra-zen na obrázku 5.4.

Souborové systémy lze připojit například zadáním následujících příkazů:

$ mount /dev/hda2 /home $ mount /dev/hda3 /usr $

Příkaz mount má dva parametry. Prvním je soubor zařízení odpovídající disku nebo diskové oblasti, na které leží připojovaný souborový systém. Druhým parametrem je adresář, pod nímž bude souborový systém připojen. Po provedení těchto příkazů vypadá obsah obou připojovaných systémů souborů tak, jako by byl součástí hierarchického stromu adresářů /home a /usr. Říkáme, že „/dev/hda2 je připojený do adresáře /home“, a podobně to platí i pro adresář /usr. Když si pak chcete prohlédnout některý ze souborových systémů, procházíte strukturou adresářů, do nichž jsou připojené, jako by to byly jakékoliv jiné adresáře. Je důležité si uvědomit rozdíl mezi souborem zařízení /dev/hda2 a adresářem /home, ke kterému je systém souborů připojen. Spe-ciální soubor zařízení umožňuje přístup k „syrovému“ obsahu pevného disku, kdežto prostřed-nictvím adresáře /home se přistupuje k souborům, které jsou na tomto disku uloženy. Adresář, ke kterému je souborový systém připojený, se nazývá *bod připojení* nebo *přípojný bod*.

Operační systém Linux podporuje mnoho typů souborových systémů. Příkaz mount se vždy pokusí rozeznat typ připojovaného systému souborů. Lze také použít přepínač -t fstyp, který přímo specifikuje typ připojovaného souborového systému. Někdy je to potřeba, protože heuris-tika programu mount nemusí vždy pracovat správně. Chcete-li například připojit disketu systému MS-DOS, zadáte příkaz:

$ mount -t msdos /dev/fd0 /floppy $

Adresář, ke kterému se souborový systém připojuje, nemusí být prázdný, ale musí existovat. Nic-méně v něm uložené soubory budou po dobu připojení nového souborového systému nedostup-né prostřednictvím svých názvů. (Soubory, které byly v době připojení otevřené, budou nadále přístupné. Soubory, na něž směřují pevné odkazy z jiných adresářů, budou přístupné prostřed-nictvím těchto odkazů.) Nehrozí žádné nebezpečí poškození těchto souborů a někdy to navíc může být i užitečné. Někteří uživatelé například rádi používají synonyma /tmp a /var/tmp. Proto si dělají symbolický odkaz adresáře /tmp do adresáře /var/tmp. Předtím než se při zavádění systému připojí souborový systém /var, je adresář /var/tmp v kořenovém souborovém systému. Po

připojení systému /var bude adresář /var/tmp v kořenovém souborovém systému nepřístup-ný. V případě, že by adresář /var/tmp v kořenovém souborovém systému neexistoval, bylo by použití dočasných souborů před připojením systému souborů /var nemožné.

Jestli nemáte v úmyslu v připojovaném souborovém systému cokoliv zapisovat, zadejte programu mount přepínač -r. Tím se vytvoří připojení pouze pro čtení. Jádro systému pak zabrání každé-mu pokusu o zápis do tohoto souborového systému a nebude ani aktualizovat časy posledního přístupu v inodech souborů. Připojení systému souborů pouze pro čtení je podmínkou u médií, na která nelze zapisovat, například u disků CD-ROM.

Pozorný čtenář si již uvědomil drobný logistický problém. Jakým způsobem se připojuje první souborový systém, tedy kořenový souborový systém (obsahující kořenový adresář celé stromové struktury), když zjevně nemůže být připojený na jiný souborový systém? Odpověď je jednoduchá
– je to kouzlo. Kořenový souborový systém se magicky připojuje při zavádění systému a lze se spolehnout na to, že bude připojený vždy. Kdyby z nějakých důvodů souborový systém root neby-lo možné připojit, systém se vůbec nezavede. Jméno souborového systému, jenž se připojuje jako kořenový, je buď zkompilované přímo v jádře systému nebo se nastavuje zavaděčem LILO, pří-padně programem rdev.

Další informace viz zdrojový kód jádra nebo Kernel Hackers' Guide – *http://tldp.org/ LDP/khg/HyperNews/get/khg.html.*

Kořenový svazek se obvykle nejdříve připojí pouze pro čtení. Pak inicializační skripty spustí pro-gram fsck, který jej zkontroluje. Když se neobjeví žádný problém, kořenový svazek se připojí znovu, a to tak, že na něj bude možno i zapisovat. Program fsck se nesmí spouštět na připoje-ném souborovém systému s možností zápisu, protože jakékoliv změny v souborovém systému, ke kterým by došlo při kontrole (a opravách) chyb v souborovém systému, způsobí *vážné* potíže. Když je kořenový souborový systém při kontrole připojený pouze pro čtení, program fsck může bez obav opravovat všechny chyby, protože operace opětovného připojení souborového systému vyprázdní všechna metadata, která má souborový systém uložená v paměti.

Na řadě systémů se při startu automaticky připojují i jiné souborové systémy. Jsou specifikované v souboru /etc/fstab. Podrobnosti týkající se formátu tohoto souboru najdete v manuálové stránce k souboru fstab. Přesné detaily procesu připojování dalších souborových systémů závisí na množství faktorů, a je-li potřeba, může je správce systému nastavit, viz kapitolu „Spouštění a zastavování systému“.

Když už není třeba mít souborový systém připojený, je možno jej odpojit příkazem umount. Pro-gram umount má jediný parametr, buďto soubor zařízení, nebo bod připojení. Například odpo-jení adresářů připojených v předchozím příkladu lze provést příkazy:

```
$ umount /dev/hda2 $ umount /usr $
```

Podrobnější instrukce, jak používat příkaz umount, najdete na manuálové stránce k tomuto pro-gramu. Je nutné vždycky odpojit disketovou mechaniku! *Nestačí jenom vytáhnout disketu z mechaniky!* Vzhledem k použití vyrovnávacích pamětí nemusí být data fyzicky zapsána, dokud není disketa odpojena. Předčasné vytažení diskety z mechaniky by mohlo způsobit poškození její-ho obsahu. Když z diskety pouze čtete, není její poškození moc pravděpodobné, ale když zapi-sujete – byť třeba jen omylem – důsledky mohou být katastrofální.

Připojování a odpojování souborových systémů vyžaduje oprávnění superuživatele, takže ho může dělat pouze uživatel root. Je tomu tak například proto, že kdyby měl kterýkoliv z uživatelů právo připojit si jednotku pružných disků na libovolný adresář, nebylo by velmi těžké připojit disketu s virem typu trójského koně „přestrojeného“ za program /bin/sh nebo jiný často používaný pří-kaz. Avšak dost často je potřeba, aby uživatelé mohli s disketami pracovat. Je několik způsobů, jak jim to umožnit:

Sdělit uživatelům heslo superuživatele. To je z hlediska bezpečnosti zjevně špatné, avšak to nejjednodušší řešení. Funguje dobře v případě, že vůbec není potřeba zabývat se zabezpečením systému. To se týká řady osobních systémů – tedy systémů, které nejsou připojeny do počítačové sítě.
Používat programy, jako je sudo, jenž umožní uživatelům používat příkaz mount. Toto je z hlediska bezpečnosti rovněž špatné řešení, avšak tímto způsobem přímo nepředáváte pri-vilegia superuživatele každému.
Umožnit uživatelům používat balík programů mtools, jenž umožňuje manipulovat se sou-borovými systémy MS-DOS bez toho, že by je bylo potřeba připojovat. Řešení funguje dobře pouze v případě, že jsou diskety systému MS-DOS vším, co budou uživatelé systé-mu potřebovat. V ostatních případech je nevyhovující.
Zapsat všechny disketové jednotky a pro ně přípustné přípojné body spolu s dalšími vhod-nými volbami do seznamu připojovaných systémů v souboru /etc/fstab.

Posledně uvedenou alternativu lze realizovat přidáním níže uvedeného řádku do souboru /etc/fstab:

```
/dev/fd0 /floppy msdos user,noauto 0 0
```

Význam jednotlivých položek (zleva doprava): soubor zařízení, které se má připojit; adresář, kam se má toto zařízení připojit; typ souborového systému; parametry; četnost zálohování (používá pro-gram dump); posledním parametrem je pořadí kontroly programem fsck (určuje pořadí, ve kterém by měly být souborové systémy prověřovány při startu systému; 0 znamená nekontrolovat).

Přepínač noauto zakáže automatické připojení při startu systému (tedy nepovolí příkazu mount -a toto zařízení připojit). Parametr

user umožní kterémukoliv uživateli připojit si souborový systém. Z důvodů bezpečnosti zamezí možnosti spouštět programy (jak běžné, tak programy s příznakem setuid) a interpretaci souborů zařízení z připojeného souborového systému. Pak si každý uživatel může připojit disketovou jednotku se souborovým systémem msdos tímto příkazem:

```
$ mount /floppy $
```

Disketu lze odpojit (a musí být odpojena) odpovídajícím příkazem umount.Chcete-li umožnit přístup k několika různým typům disket, musíte zadat několik přípojných bodů.Nastavení pro každý přípojný bod mohou být různá. Například přístup k disketám s oběma typysouborových systémů – MS-DOS i ext2 – byste umožnili přidáním těchto řádků do souboru/etc/fstab:

```
/dev/fd0 /dosfloppy msdos user,noauto 0 0 /dev/fd0 /ext2floppy ext2 user,noauto 0 0
```

Volba auto ve sloupci filesystem umožňuje příkazu mount zeptat se na souborový systém a zkusit
jej určit. Tato volba nefunguje u všech souborových systémů, u běžných však funguje spolehlivě. U souborových systémů MS-DOS (nejenom na disketách) budete pravděpodobně vyžadovat ome-zený přístup s využitím systémových parametrů uid, gid a umask. Ty jsou podrobně popsané v manuálové stránce příkazu mount. Následkem neopatrnosti při připojování souborového syste-mu MS-DOS může být totiž to, že kterýkoliv uživatel získá oprávnění číst v připojeném systému souborů kterékoliv soubory, což není moc dobré.

Kontrola integrity souborového systému programem fsck

Souborové systémy jsou poměrně složité struktury a tím také mají sklony k chybovosti. Správnost a platnost souborového systému se kontroluje příkazem fsck. Lze jej nastavit tak, aby při kontro-le automaticky opravoval méně závažné chyby a upozorňoval uživatele na výskyt chyb, které nelze odstranit. Naštěstí je kód implementující souborové systémy odladěný velmi dobře, takže problémy vznikají jen zřídka a jsou obvykle zapříčiněny výpadkem napájecího napětí, selháním technického vybavení nebo chybou obsluhy, například nesprávným vypnutím systému.

Většina systémů je nastavena tak, že spouští program fsck automaticky při zavádění systému, takže jakékoliv chyby jsou detekovány (a většinou i odstraněny) předtím, než systém přechází do běžného pracovního režimu. Používáním poškozeného systému souborů se totiž jeho stav obvyk-le ještě více zhoršuje. Jsou-li v nepořádku datové struktury souborového systému, práce se systé-mem je pravděpodobně poškodí ještě víc, důsledkem čehož mohou být ztráty dat většího rozsa-hu. Kontrola velkých souborových systémů programem fsck chvíli trvá. Když se ale systém vypí-ná správně, chyby souborových systémů se vyskytují jenom velmi zřídka. Lze pak využít několi-ka triků, díky kterým se lze kontrole (velkých) souborových systémů vyhnout, není-li to nutné. První trik spočívá v tom, že existuje-li soubor /etc/fastboot, neprovádí se žádná kontrola. Druhý: Souborový systém ext2 má ve svém superbloku speciální příznak, jehož hodnota je nasta-vena podle toho, jestli byl souborový systém po předchozím připojení odpojen správně či niko-
liv. Podle tohoto příznaku pozná pak program e2fsck (verze příkazu fsck pro souborový systém ext2), jestli je nutné provádět kontrolu tohoto systému souborů či nikoliv. V případě, že podle hodnoty příznaku byl daný systém souborů odpojen korektně, kontrola se neprovádí. Předpoklá-dá se, že řádné odpojení systému zároveň znamená, že v souborovém systému nevznikly žádné defekty. To, zda se při proceduře zavádění systému vynechá proces kontroly systému souborů v případě, že soubor /etc/fastboot existuje, záleží na nastavení spouštěcích skriptů systému. Trik s příznakem souborového systému ext2 ale funguje vždy, když systém souborů kontrolujete programem e2fsck. Kdybyste totiž chtěli programu e2fsck přikázat, aby prověřil i korektně odpo-jený systém souborů, museli byste tento požadavek explicitně zadat odpovídajícím přepínačem. (Podrobnosti uvádí manuálová stránka programu e2fsck.)

Automatické prověřování správnosti souborových systémů se provádí jenom u systémů souborů, které se připojují automaticky při startu operačního systému. Manuálně lze program fsck použít pro kontrolu jiných souborových systémů, například na disketách.

Najde-li program fsck neodstranitelné chyby, připravte se na to, že budete potřebovat buď hlu-boké znalosti obecných principů fungování souborových systémů a konkrétního typu poškoze-ného souborového systému zvlášť nebo dobré zálohy. Druhou možnost lze jednoduše (ačkoli někdy dost pracně) zajistit. Nemáte-li potřebné „know-how“ sami, mohou první alternativu v některých případech zajistit vaši známí, dobrodinci z diskusních skupin o Linuxu na Internetu, popřípadě jiný zdroj technické podpory. Rádi bychom vám poskytli víc informací týkajících se této problematiky, bohužel nám v tom brání nedostatek podrobných znalostí a zkušeností. Jinak uži-tečný by pro vás mohl být program debugfs, jehož autorem je Theodore T’so.
Program fsck může běžet pouze na odpojených souborových systémech, v žádném případě ne na připojených (s výjimkou kořenového souborového systému připojeného při zavádění systému pouze pro čtení). To proto, že program přistupuje k „syrovému“ disku, a tak může modifikovat systém souborů bez toho, že by využíval operační systém. Když se vám podaří operační systém tímto způsobem „zmást“, téměř určitě můžete očekávat problémy.

Kontrola chyb na disku programem badblocks

Je vhodné pravidelně kontrolovat výskyt vadných bloků na disku. Dělá se to příkazem badblocks. Jeho výstupem je seznam čísel všech vadných bloků, na které program narazil. Tímto seznamem pak můžete „nakrmit“ program fsck, který podle něj provede

záznamy do datových struktur sou-borového systému, takže operační systém se nepokouší využívat vadné bloky pro ukládání dat. V následujícím příkladu je naznačen celý postup:

$ badblocks /dev/fd0H1440 1440 > bad-blocks $ fsck -t ext2 -l bad-blocks /dev/fd0H1440 Parallelizing fsck version 0.5a (5-Apr-94) e2fsck 0.5a, 5-Apr-94 for EXT2 FS 0.5, 94/03/10 Pass 1: Checking inodes, blocks, and sizes <zkráceno> /dev/fd0H1440: ***** FILE system WAS MODIFIED ***** /dev/fd0H1440: 11/360 files, 63/1440 blocks $

Je-li v seznamu vadných bloků uveden blok, který je již některým ze souborů využíván, program e2fsck se pokusí tento sektor přemístit na jiné místo disku. Když je blok skutečně vadný a nejde jenom o logickou chybu vzniklou při zápisu, bude obsah souboru s největší pravděpodobností poškozený.

Boj s fragmentací

Když se soubor ukládá na disk, nemůže být vždy zapsán do po sobě jdoucích bloků. Soubor, jenž není uložen do souvislé řady za sebou jdoucích bloků, je *fragmentovaný*. Načtení takového sou-boru pak trvá déle, protože čtecí hlava disku se musí při čtení víc pohybovat. Proto je žádoucí zamezit fragmentaci souborů, i když pro systémy s velkou vyrovnávací pamětí a dopředným čte-ním nepředstavuje až tak závažnou komplikaci.

Moderní linuxové souborové systémy minimalizují fragmentaci tak, že udržují všechny bloky souboru blízko u sebe, i když nemohou být uloženy do sousedících sektorů. Některé systémy, např. ext3, přidělují ty volné bloky, které jsou nejblíže ostatním blokům v souboru. Fragmentací se tedy není nutno zabývat.

Pro systém MS-DOS existuje celá řada defragmentačních programů. Ty přesouvají bloky v soubo-rovém systému tak, aby fragmentaci odstranily. V ostatních souborových systémech se musí defragmentace dělat tak, že se vytvoří záloha souborového systému, který se znovu vytvoří, a ze zálohy se obnoví soubory. Doporučení zálohovat souborový systém před jeho defragmentací se týká všech souborových systémů, protože v průběhu procesu defragmentace může dojít k poško-zení systému souborů i k dalším chybám.

Další nástroje pro všechny souborové systémy

Existují i některé další nástroje užitečné pro správu souborových systémů. Program df ukazuje volný diskový prostor v jednom či několika souborových systémech. Program du ukazuje, kolik diskového prostoru zabírá adresář a všechny soubory v něm uložené. Lze jej s výhodou využít při „honu“ na uživatele, kteří zabírají svými (mnohdy zbytečnými) soubory nejvíc místa na disku. Oba nástroje mají manuálové stránky s podrobným popisem mnoha funkcí.

Program sync zapíše všechny neuložené bloky z vyrovnávací paměti (viz kapitolu „Vyrovnávací paměť“) na disk. Jen zřídkakdy je potřeba zadávat jej ručně, automaticky to totiž dělá démon update. Má to význam především v případě nečekaných událostí, například když je proces upda-te nebo jeho pomocný proces bdflush nečekaně ukončen nebo v případě, že musíte *ihned* vypnout napájení a nemůžete čekat, než se opět spustí program update. Opět upozorňujeme na manuálové stránky. Váš nejlepší přítel je v man Linuxu. Užitečný je také jeho bratranec apropos pro případ, že byste nevěděli, jaký příkaz použít.

Další nástroje pro souborový systém ext2/ext3

Kromě programu, kterým se tento systém souborů vytváří (mke2fs), a programu pro kontrolu jeho integrity (e2fsck), jež jsou přístupné přímo z příkazové řádky, případně přes koncové pro-gramy nezávislé na typu souborového systému, existují pro souborový systém ext2 některé další nástroje, jež mohou být při správě systému užitečné.

Program tune2fs umí upravit parametry souborového systému. Uvedeme některé z těch význam-nějších parametrů:

Maximální počet připojení, po němž program e2fsck provede kontrolu souborového systé-mu bez ohledu na to, že je nastaven příznak korektního odpojení. U systémů, které jsou určeny k vývoji nebo testování, je rozumné tento limit snížit.
Maximální čas mezi kontrolami integrity. Příkaz e2fsck hlídá maximální periodu mezi dvěma kontrolami a provede opět kontrolu i v případě, že je nastaven příznak korektního vypnutí a souborový systém nebyl připojen vícekrát, než je povoleno. Opakované kontro-ly lze zakázat.
Počet bloků vyhrazených pro superuživatele. Souborový systém ext2 rezervuje některé bloky pro superuživatele. Když se pak souborový systém jako celek zaplní, není potřeba nic mazat a systém lze v omezené míře spravovat. Rezervovaný počet bloků je implicitně nastaven na 5 %, což u většiny disků stačí k tomu, aby se zamezilo jejich přeplnění. V případě disket nemá tato rezervace smysl.

Více informací najdete v manuálové stránce programu tune2fs. Program dumpe2fs vypisuje informace o souborovém systému typu ext2, většinou z jeho super-bloku. Dále uvidíte příklad výstupu tohoto programu. Některé z těchto informací jsou ryze tech-nické a vyžadují hlubší pochopení problematiky fungování tohoto souborového systému. Většina údajů je ale snadno pochopitelná i pro správce-laiky.

dumpe2fs dumpe2fs 1.32 (09-Nov-2002) Filesystem volume name: / Last mounted on: not available Filesystem UUID: 51603f82-68f3-4ae7-a755-b777ff9dc739 Filesystem magic number: 0xEF53 Filesystem revision #: 1 (dynamic) Filesystem features: has_journal filetype needs_recovery sparse_super Default mount options: (none) Filesystem state: clean Errors behavior: Continue Filesystem OS type: Linux Inode count: 3482976 Block count: 6960153 Reserved block count: 348007 Free blocks: 3873525 Free inodes: 3136573 First block: 0 Block size: 4096 Fragment size: 4096 Blocks per group: 32768 Fragments per group: 32768 Inodes per group: 16352 Inode blocks per group: 511 Filesystem created: Tue Aug 26 08:11:55 2003 Last mount time: Mon Dec 22 08:23:12 2003 Last write time: Mon Dec 22 08:23:12 2003 <zkráceno>

Program debugfs je nástroj pro ladění souborového systému. Umožňuje přímý přístup k struktu-rám dat souborového systému uloženým na disku, a lze jej proto použít při opravách disků poško-zených natolik, že to nezvládne program fsck. Lze jej též využít k obnově smazaných souborů. Pokud chcete použít program debugfs, je velmi důležité, abyste skutečně věděli, co děláte. V pří-padě, že zcela neporozumíte některé jeho funkci, se totiž může stát, že všechna svá data zničíte.

Programy dump a restore lze použít při zálohování souborového systému ext2. Jsou to specific-ké verze tradičních nástrojů. Víc informací o zálohování najdete v kapitole „Jak je důležité míti zálohy“.

Disky bez souborových systémů

Ne na všech discích nebo diskových oblastech se vytváří souborové systémy. Například disková odkládací oblast nebude používat žádný souborový systém. Dalším příkladem jsou diskety, jejichž mechaniky se mnohdy používají pro emulaci páskové jednotky. Program tar nebo jiný archivač-ní nástroj pak zapisuje přímo na „syrový“ disk bez souborového systému. Zaváděcí diskety systé-mu Linux rovněž nemají souborový systém, pouze holé jádro systému.

Skutečnost, že se na disku (disketě) nevytvoří souborový systém, má výhodu v tom, že lze využít větší část kapacity disku, protože každý souborový systém má vždy nějakou režii. Navíc lze takto dosáhnout větší kompatibility disků s jinými operačními systémy, například soubor s formátem, jenž používá program tar, má stejnou strukturu ve všech operačních systémech, kdežto souboro-vé systémy samotné jsou ve většině operačních systémů různé. Další výhodou je, že disky bez sou-borových systémů lze v případě potřeby použít velmi rychle (odpadá operace vytváření a valida-ce souborového systému). Zaváděcí diskety Linuxu také nutně nemusí obsahovat souborový systém, ačkoliv to možné je.

Dalším z důvodů, proč používat „syrová“ zařízení, je možnost vytvořit přesný zrcadlový obraz – kopii disku. Když například disk obsahuje částečně poškozený souborový systém, je vhodné před-tím, než se pokusíte chybu opravit, udělat přesnou kopii poškozeného disku. Pak není problém začít znova v případě, že při neúspěšném pokusu opravit chybu poškodíte systém souborů ještě víc. Jedním ze způsobů, jak zrcadlovou kopii disku udělat, je použít program dd:

$ dd if=/dev/fd0H1440 of=floppy-image 2880+0 records in
2880+0 records out $ dd if=floppy-image of=/dev/fd0H1440 2880+0 records in 2880+0 records out $

První příkaz dd uloží přesný obraz diskety do souboru floppy-image, druhý zapíše tento obraz na další disketu. (Samozřejmě se předpokládá, že uživatel před zadáním druhého příkazu diskety v mechanice vyměnil. Jinak by byly tyto příkazy k ničemu.)

Přidělování diskového prostoru

Způsoby rozdělování disku na diskové oblasti

Rozdělit disk na diskové oblasti tím nejlepším možným způsobem není vůbec jednoduché. A co je nejhorší – neexistuje univerzálně správný způsob, jak toho dosáhnout. Celý problém totiž komplikuje příliš mnoho různých faktorů.

„Tradiční“ variantou je mít (relativně) malý kořenový souborový systém a oddělené části pro sou-borové systémy, jako např. /usr nebo /home>. Vytvoříme-li zvláštní kořenový souborový systém, který je malý a málo používaný, pravděpodobnost jeho poškození při havárii systému je menší, a obnova systému je tedy snazší. Kořenový souborový systém by se neměl zaplnit, aby nezpůso-bil havárii systému.

Když si vytváříte schéma rozdělení na oblasti, je důležité si uvědomit určité skutečnosti. Pro adre-sáře /bin, /etc, /dev, /initrd, /lib a /sbin nemůžete vytvořit samostatné oblasti. Obsah těch-to adresářů se využívá při zavádění systému, a proto musí být vždy součástí oblasti /.

Také je vhodné vytvořit samostatné oblasti pro adresáře /var a /tmp. Důvodem je skutečnost, že oba adresáře obvykle obsahují data, která se průběžně mění. Pokud byste pro tyto souborové systémy nevytvořili samostatné oblasti, riskujete, že žurnál zaplní oblast /.

Příklad serverových oblastí:

Filesystem Size Used Avail Use% Mounted on /dev/hda2 9.7G 1.3G 8.0G 14% / /dev/hda1 128M 44M 82M
34% /boot /dev/hda3 4.9G 4.0G 670M 86% /usr /dev/hda5 4.9G 2.1G 2.5G 46% /var /dev/hda7 31G 24G
5.6G 81% /home /dev/hda8 4.9G 2.0G 670M 43% /opt

Problém několika samostatných diskových oblastí je v tom, že rozdělují celkové množství volné
ho diskového prostoru na mnoho malých částí.
Ukázka oblastí na domácím počítači:

Filesystem Size Used Avail Use% Mounted on /dev/sda1 6,7G 4,3G 2,1G 68% / /dev/sda6 85G 74G 12G
87% /home

Správce logických svazků (LVM)

Pomocí LVM může administrátor vytvářet logické disky a podle potřeby tak reagovat na požadav-ky na prostor na disku. Řeší to například výše uvedený problém s rozdělením disku na mnoho malých částí, které lze pomocí LVM spojovat do logických svazků a podle potřeby přeskupovat.
Administrátor pomocí linuxového správce LVM nejprve vytvoří oblasti typu 0x8e. Tyto *fyzické oblasti* pak přidá ke *skupině svazků* a rozdělí je na úseky, které nazýváme *fyzická rozšíření sku-piny svazků*. Tato rozšíření seskupí do *logických svazků*. Logické svazky je možné formátovat stej-ně jako fyzické oblasti. Podstatný rozdíl spočívá v tom, že k logickému svazku můžeme přidávat další rozšíření.

Úplný popis LVM překračuje rozsah této knihy. Vynikající zdroj informací o LVM naleznete na *http://www.tldp.org/HOWTO/LVM-HOWTO.html.*

Nároky na diskový prostor

Distribuce Linuxu, kterou budete instalovat, vám obvykle nějakým způsobem sdělí, kolik diskového prostoru je potřeba pro různé konfigurace operačního systému. Programy, které se instalují dodatečně, se budou většinou chovat stejně. Díky tomu si můžete udělat představu o nárocích na diskový prostor a naplánovat si jeho rozdělení. Měli byste se ale připravit i na budoucnost a vyhradit si nějaké místo navíc pro věci, na které si vzpomenete později.

Prostor, který byste měli vyčlenit pro soubory uživatelů systému, závisí na tom, co budou dělat. Většina lidí totiž obvykle spotřebuje tolik diskového prostoru, kolik je jenom možné. Nicméně množství, které jim skutečně stačí, je velmi různé. Někteří uživatelé vystačí s psaním textů a spo-kojeně přežijí i s několika megabajty. Jiní dělají složitou editaci grafických souborů a budou potře-bovat gigabajty volného prostoru.

Mimochodem, když budete při odhadech nároků na diskový prostor srovnávat velikosti souborů v kilobajtech nebo megabajtech a diskový prostor v megabajtech, uvědomte si, že tyto dvě jed-notky mohou být různé. Někteří výrobci disků rádi tvrdí, že kilobajt je 1 000 bajtů a megabajt je 1 000 kilobajtů, kdežto zbytek počítačového světa používá pro oba koeficienty číslo 1 024. Proto váš 345MB pevný disk bude mít ve skutečnosti pouze 330 MB.

O přidělování odkládacího prostoru hovoříme v kapitole „Vytvoření odkládacího prostoru na disku“.

Příklad rozvržení diskového prostoru

Původně jsem používal pevný disk o velikosti 10 GB, nyní jsem přešel na 30GB disk. Vysvětlím,
jak a proč jsem tyto disky rozdělil na oblasti. Nejdříve jsem si vytvořil oblast /boot o velikosti 128 MB. To je víc, než budu potřebovat, rezer-va je dostatečná. Samostatnou oblast /boot jsem vytvořil proto, abych měl jistotu, že se tento sou-borový systém nikdy nepřeplní a systém půjde v každém případě zavést. Poté jsem vytvořil 5GB oblast /var. Vzhledem k tomu, že do souborového systému /var se zapisují logy a pošta, chtěl jsem jej mít oddělený od kořenové oblasti. (Dříve se mi totiž stalo, že logy přes noc vzrostly tak, že zaplnily celý kořenový souborový systém.) Dále jsem vytvořil 15GB oblast /home. Ta se hodí při havárii systému. Pokud bych někdy musel reinstalovat Linux, mohu říct instalačnímu progra-mu, aby tuto oblast neformátoval, nýbrž ji rovnou připojil, čímž nedojde ke ztrátě dat. Nakonec, vzhledem k tomu, že jsem měl 512 MB RAM, vytvořil jsem si 1 024 MB (resp. 1 GB) velkou odklá-dací oblast. Na kořenový systémový soubor mi tak zbylo kolem 9 GB. Na svém starém 10GB disku jsem si vytvořil 8GB oblast /usr a ponechal 2 GB nevyužité. To pro případ, že bych v budouc-nosti potřeboval další prostor.

Nakonec vypadala má tabulka oblastí takto:

9 GB	kořenový souborový systém
1 GB	odkládací oblast
5 GB	souborový systém /var
15 GB	souborový systém /home
8 GB	souborový systém /usr
2 GB	dočasná oblast

Moje diskové oblasti

Zvětšování diskového prostoru pro Linux

Zvětšení diskového prostoru vyhrazeného pro Linux je jednoduché. Především v případě, že se instalují nové pevné disky (popis instalace disků jde ale nad rámec našeho návodu). Je-li to nutné, je potřeba disky zformátovat. Pak se podle výše uvedeného postupu vytvoří diskové oblasti a sou-borový systém a přidají se odpovídající řádky do souboru /etc/fstab kvůli tomu, aby se nové disky připojovaly automaticky.

Tipy, jak ušetřit místo na disku

Nejlepší způsob, jak ušetřit diskový prostor, je vyvarovat se instalování nepotřebných programů. Většina distribucí Linuxu při instalaci nabízí možnost výběru balíků programů, které se mají insta-lovat. Podle této nabídky byste měli být schopni analyzovat své potřeby a pak pravděpodobně zji-stíte, že většinu z nich nebudete potřebovat. Tím se ušetří hodně místa na disku, protože některé z programů a aplikačních balíků jsou dost objemné. I když zjistíte, že budete některou aplikaci nebo i celý balík programů potřebovat, určitě nebudete muset instalovat všechny jejich součásti. Obvykle není potřeba instalovat například on line dokumentaci stejně jako některé ze souborů Elisp pro GNU verzi programu Emacs, některé z fontů pro X Window či některé knihovny pro-gramovacích jazyků.

V případě, že nemůžete některé balíky programů trvale odinstalovat, můžete využít kompresi. Komprimační programy jako gzip nebo zip zabalí (a rozbalí) jednotlivé soubory, případně celé skupiny souborů. Program gzexe transparentně komprimuje a dekomprimuje programy tak, že to uživatel v běžném provozu nepostřehne (nepoužívané programy se komprimují a rozbalí se, až když jsou potřeba). V současné době se testuje systém DouBle, který transparentně komprimuje všechny soubory v souborovém systému. (Znáte-li program Stacker pro MS-DOS, princip je podobný.)

Jiný způsob, jak uspořit paměť, je věnovat zvláštní pozornost formátování oblastí. Většina moderních souborových systémů umožňuje zadávat velikost bloku. Velikost bloku udává velikost, kterou bude systém používat pro čtení a zápis dat. Větší bloky zvýší výkon disku při práci s velkými soubory, například s databázemi. Důvodem je skutečnost, že disk se může déle věnovat čtení nebo zápisu, než začne hledat další blok.

Uvědomte si ale, že zatímco výše uvedené rady mohou uspořit desítky či stovky megabytů (ve výjimečných případech možná více), tak kapacita dnes běžně dodávaných disků je ve stovkách gigabajtů. Nabízí se otázka, zda je při takové konfiguraci podobná úspora místa vůbec nutná.

Správa paměti

Minnet, jag har tappat mitt minne, är jag svensk eller finne,
kommer inte ihĺg. (Bosse Österberg)

Švédská pijácká písnička, přibližně: *„Paměť, ztratil jsem paměť,*
jsem Švéd nebo Fin, nemůžu si vzpomenout.“

V této kapitole jsou popsány hlavní rysy systému správy paměti operačního systému Linux, tedy subsystémy virtuální paměti a diskové vyrovnávací paměti. Kapitola popisuje jejich účel, funkce a další podrobnosti, které správce systému musí mít na paměti.

Co je virtuální pamě?

Operační systém Linux podporuje *virtuální paměť*, to znamená, že používá disk jako rozšíření paměti RAM. Tím se efektivní velikost využitelné paměti odpovídajícím způsobem zvětší. Jádro systému zapisuje obsah právě nevyužívaných paměťových bloků na pevný disk a paměť se tak může využívat pro jiné účely. Když pak přijde požadavek na její původní obsah, bloky z disku se načtou zpět do paměti. To vše probíhá z pohledu uživatele zcela transparentně. Programy běžící pod Linuxem vidí pouze, že je k dispozici hodně paměti, a nestarají se o to, že její část je občas uložená na disku. Přirozeně, čtení a zápis na pevný disk je pomalejší (zhruba o tři řády) než vyu-žití reálné paměti, takže programy neběží tak rychle. Část disku, která se využívá jako virtuální paměť, se nazývá *odkládací prostor – swap*.

Linux může použít jako odkládací prostor nejen normální soubor uložený v souborovém systému, ale i diskové oblasti. Předností

diskových oblastí je rychlost, výhodou odkládacího souboru je jed-nodušší možnost změny celkové velikosti odkládacího prostoru. Není přitom totiž potřeba měnit rozdělení celého pevného disku, kdy navíc hrozí nutnost kompletní reinstalace systému. Víte-li, jak velký odkládací prostor budete potřebovat, zvolte odkládací prostor na zvláštní diskovou oblast. Pokud si nároky nejste zcela jisti, zvolte nejdřív odkládací prostor v souboru. Když bude-te systém nějakou dobu používat, budete schopni odhadnout, kolik odkládacího prostoru skuteč-ně potřebujete. Až budete mít ohledně předpokládané velikosti požadovaného odkládacího pro-storu jasno, vytvoříte pro něj zvláštní diskovou oblast.

Měli byste rovněž vědět, že Linux umožňuje využívat současně několik odkládacích oblastí, případně několik odkládacích souborů. Takže když potřebujete pouze příležitostně větší množství odkládacího prostoru, je lepší (místo trvale vyhrazené rezervy) nastavit další soubor navíc.

Poznámka k terminologii používané v oblasti operačních systémů: V odborných kruzích se obvykle rozlišuje mezi odkládáním (anglicky *swapping*), tedy zapsáním celého procesu do odklá-dacího prostoru na disku, a stránkováním (anglicky *paging*), tedy zapisováním pouhých částí pevné velikosti (obvykle několik kilobajtů) najednou. Stránkování je obecně výkonnější a je to metoda, kterou používá i operační systém Linux. Tradiční terminologie systému Linux ale používá pojem odkládání (swapping).

Vytvoření odkládacího prostoru na disku

Odkládací soubor je běžný soubor a není pro jádro systému ničím zvláštní. Jediná vlastnost, která má pro jádro význam, je, že odkládací soubor nemá díry a že je připraven pro použití programem mkswap. Musí být navíc (z důvodů implementace) uložen na lokálním disku, takže nemůže být uložen v souborovém systému, který je připojen pomocí NFS.

Zmínka o dírách je důležitá. Odkládací soubor rezervuje určitý diskový prostor, takže jádro systé-mu pak může rychle odložit stránku paměti bez toho, že by muselo absolvovat celou proceduru alokace diskového prostoru, která se používá pro běžný soubor. Jádro využívá pouze ty sektory, které byly odkládacímu souboru skutečně přiděleny. Protože díra v souboru znamená, že tomuto místu souboru nejsou přiděleny žádné diskové sektory, nemohlo by je jádro dost dobře využít.

Jeden ze způsobů, kterým lze vytvořit odkládací soubor bez prázdných míst, je tento:

```
$ dd if=/dev/zero of=/extra-swap bs=1024 count=1024 1024+0 records in 1024+0 records out $
```

kde /extra-swap je jméno odkládacího souboru, jehož velikost je uvedena za parametrem count=. Ideální je zvolit velikost jako násobek 4, protože jádro systému zapisuje do odkládacího prostoru *stránky paměti*, které jsou 4 kilobajty velké. Nebude-li velikost násobkem 4, může být posledních pár kilobajtů nevyužitých.

Samostatná odkládací disková oblast rovněž není ničím neobvyklým. Vytvoří se stejně jako každá jiná disková oblast. Jediným rozdílem je, že se používá jako holé zařízení, tedy bez souborového systému. Je dobré označit ji jako typ 82 („Linux swap“). I když to jádro systému striktně nevyžaduje, vnese to do seznamu diskových oblastí řád.
Poté co vytvoříte odkládací soubor nebo diskovou oblast pro odkládací prostor, je potřeba zapsat na jejich začátek signaturu, která obsahuje některé administrativní informace a s níž jádro pracuje. Provede se to příkazem mkswap tímto způsobem:

```
$ mkswap /extra-swap 1024 Setting up swapspace, size = 1044480 bytes $
```

Odkládací prostor zatím není využitý. Sice existuje, ale jádro systému jej jako virtuální paměť zatím
nezná. Při zadávání příkazu mkswap byste měli být velice opatrní, protože program nekontroluje, zda se soubor nebo disková oblast nevyužívá k jiným účelům. *Příkazem mkswap proto můžete lehce přepsat důležité soubory nebo celé diskové oblasti!*
Naštěstí budete tento příkaz potřebovat, pouze když instalujete operační systém.

Systém správy paměti v Linuxu omezuje velikost odkládacího prostoru na 2 GB. V nových jádrech můžete ovšem používat až 32 takových prostorů, což je celkem 64 GB. Maximální velikost i počet odkládacích prostorů ale mohou být omezeny nebo naopak zvětšeny hardwarovou architekturou či verzí jádra, více viz man mkswap.

Využívání odkládacího prostoru

Využívání nově vytvořeného odkládacího prostoru lze zahájit příkazem swapon. Příkaz sdělí jádru systému, že odkládací prostor, jehož úplná cesta se zadává jako parametr příkazu, lze od této chví-le používat. Takže když budete chtít začít využívat dočasný soubor jako odkládací prostor, zadá-te tento příkaz:

```
$ swapon /extra-swap $
```

Odkládací prostory lze využívat automaticky poté, co budou zapsány v souboru /etc/fstab, například:

```
/dev/hda8 none swap sw 0 0 /swapfile none swap sw 0 0
```

Spouštěcí skripty vykonávají příkaz swapon -a, jenž zahájí odkládání do všech odkládacích pro-storů uvedených v souboru

/etc/fstab. Takže příkaz swapon se obvykle používá, jenom když je potřeba použít odkládací prostor navíc.

Příkazem free lze monitorovat využívání odkládacích prostorů. Příkaz zobrazí celkové množství odkládacího prostoru, který je v systému využíván:

$ free total used free shared buffers Mem: 15152 14896 256 12404 2528 -/+ buffers: 12368 2784 Swap: 32452 6684 25768 $

V prvním řádku výstupu (Mem:) se zobrazuje velikost fyzické paměti. Sloupec total neukazujevelikost fyzické paměti, kterou využívá jádro systému, ta má obvykle asi jeden megabajt. Ve sloup-ci used je zobrazeno množství využívané paměti (v druhém řádku je vynechána velikost vyrov-návací paměti). Sloupec free udává celkové množství nevyužité paměti. Sloupec shared ukazujepaměť sdílenou několika procesy – platí čím více, tím lépe. Ve sloupci buffers je zobrazena aktu-ální velikost vyrovnávací paměti.

V posledním řádku (Swap:) jsou analogické informace pro odkládací prostor. Když jsou v tomto

řádku samé nuly, není odkládací prostor systému aktivovaný.Stejné informace lze získat příkazem top nebo z údajů v souborovém systému proc, přesnějiv souboru /proc/meminfo. Obvykle je ale dost obtížné získat informace o využití jednoho kon-krétního odkládacího prostoru.

Odkládací prostor lze vyřadit z činnosti příkazem swapoff. Příkaz pravděpodobně využijetepouze pro vyřazení dočasných odkládacích prostorů. Všechny stránky, které jsou uloženy v odklá-dacím prostoru, se po zadání příkazu swapoff nejdříve načtou do paměti. Když není dostatekfyzické paměti, do které by se načetly, budou uloženy do některého z jiných odkládacích prosto-rů. Pokud však není ani dostatek virtuální paměti na odložení všech načítaných stránek odkláda-cího prostoru, jenž má být vyřazen z činnosti, začnou problémy. Po delší době by se operačnísystém měl zotavit, ale mezitím bude prakticky nepoužitelný. Proto byste měli předtím, než vyřa-díte některý odkládací prostor z činnosti, zkontrolovat (například příkazem free), jestli máte dosta-tek volné paměti.
Všechny odkládací prostory, které se aktivují automaticky příkazem swapon -a, lze vyřadit z čin-nosti příkazem swapoff -a. Příkaz vyřadí z činnosti odkládací prostory uvedené v souboru /etc/fstab. Odkládací prostory přidané ručně zůstanou nadále v činnosti.

Někdy mohou nastat situace, že se využívá příliš mnoho odkládacího prostoru, i když má systém dostatek volné fyzické paměti. Může se to stát například v situaci, kdy jsou v jednom okamžiku velké nároky na virtuální paměť, ale po chvíli je ukončen některý větší proces, který využívá větší část fyzické paměti, a ten paměť uvolní. Odložená data se ale nenačítají do paměti automaticky a zůsta-nou uložena na disku až do doby, než budou potřeba. Fyzická paměť by tak mohla zůstat dost dlou-ho volná, nevyužitá. Není potřeba se tím znepokojovat, ale je dobré vědět, co se v systému děje.

Sdílení odkládacího prostoru s jinými operačními systémy

Virtuální paměť používá mnoho operačních systémů. Vzhledem k tomu, že každý ze systémů vyu-žívá svůj odkládací prostor, jenom když běží (tedy nikdy ne několik systémů současně), odkláda-cí prostory ostatních operačních systémů pouze zabírají místo na disku. Pro operační systémy by bylo efektivnější sdílet jediný odkládací prostor. I to je možné, ale je potřeba si s tím pohrát. V textu *Tips-HOWTO* na adrese *http://www.tldp.org/HOWTO/Tips-HOWTO.html* je uvedeno něko-lik praktických rad, jak implementovat sdílení odkládacího prostoru.

Přidělování odkládacího prostoru

Možná někdy narazíte na radu, že máte přidělovat dvakrát tolik odkládacího prostoru, než máte fyzické paměti. To je ovšem pouhá pověra[1]. Uvádíme proto správný postup:
Zkuste odhadnout, jaké budete mít nároky na paměť, tedy největší množství paměti, které bude te pravděpodobně v jednom okamžiku potřebovat. To je dané součtem paměťových nároků všech programů, které poběží současně.

Když například chcete, aby běželo grafické uživatelské rozhraní X Window, měli byste mu přidě-lit asi 8 MB paměti. Kompilátor gcc vyžaduje několik megabajtů (kompilace některých souborů by mohla mít neobvykle velké nároky na paměť, jež by mohly dosáhnout až několika desítek mega-bajtů, ale běžně by měly stačit zhruba čtyři megabajty). Jádro samotné bude využívat asi jeden megabajt paměti, běžné interprety příkazů a jiné menší nástroje několik stovek kilobajtů (řekně-me asi jeden megabajt dohromady). Není potřeba být v odhadech úplně přesný, stačí udělat velmi hrubý odhad, ale ten by měl být spíše pesimistický.

Je důležité si uvědomit, že když bude systém současně používat více uživatelů, budou paměť RAM potřebovat všichni. Avšak budou-li současně ten samý program používat dva uživatelé, celková

[1] Poznámka českého vydavatele: Autoři to zřejmě nemysleli úplně vážně. Existuje hned několik dobrých důvodů, proč tento vzorec

použít, a nejen proto se dnes hojně používá v instalátorech různých distribu-cí. Zaprvé je velmi jednoduchý a pro desktop zcela postačující – srovnejte jej s náročností dále uvádě-ného postupu, kde se autoři snaží šetřit megabyty, což je u dnešních disků téměř směšný objem. Navíc operují se starými údaji o nárocích jednotlivých programů, ve skutečnosti jsou dnes větší a musíte si je zjistit sami. Případné předimenzování odkládacího oddílu zase tolik nevadí, na disku se „skoro ztratí“ a navíc minimalizujete riziko vlastního špatného odhadu – problém klasického odkládacího oddílu je totiž v tom, že se nedá tak jednoduše zvětšit. A posledním důvodem jsou přenosné počítače, kde se při uspá-vání na disk zapisuje aktuální stav systému právě do odkládacího oddílu. Proto je nutné, aby měl veli-kost větší než operační paměť (opět se doporučuje dvojnásobek). Pro servery byste určitě měli použít uvedený postup, protože se může stát, že nároky na odkládací prostor budou ještě mnohem větší.

potřeba paměti obvykle nebude dvojnásobná, protože stránky kódu a sdílené knihovny budouv paměti pouze jednou.Pro správný odhad nároků na paměť jsou užitečné příkazy free a ps.

K odhadu podle předchozího kroku připočítejte nějakou rezervu. To proto, že odhady paměťových nároků programů budou velmi pravděpodobně nedostatečné – je možné, že na některé aplikace, které budete chtít používat, zapomenete. Takto budete mít jistotu, že pro tento případ máte nějaké to místo navíc. Mělo by stačit pár megabajtů. (Je lepší vyčle-nit příliš mnoho než moc málo odkládacího prostoru. Ale není potřeba to přehánět a alo-kovat celý disk, protože nevyužitý odkládací prostor zbytečně zabírá místo. V dalších čás-tech této kapitoly bude uvedeno, jak přidat další odkládací prostor.) Vzhledem k tomu, že se lépe počítá s celými čísly, je dobré hodnoty zaokrouhlovat směrem nahoru, řádově na další celé megabajty.
Na základě těchto výpočtů budete vědět, kolik paměti budete celkem potřebovat. Takže když odečtete velikost fyzické paměti od celkových nároků na paměť, dozvíte se, kolik odkláda-cího prostoru musíte celkem vyčlenit. (U některých verzí Unixu se musí vyčlenit i paměťový prostor pro obraz fyzické paměti, to znamená, že velikost paměti vypočítaná podle kroku 2 představuje skutečné nároky na odkládací prostor a neodečítá se velikost fyzické paměti.)
Je-li vypočítaná velikost odkládacího prostoru o hodně větší než dostupná fyzická paměť (více než dvakrát), měli byste raději více investovat do fyzické paměti. Jinak bude výkon systému příliš nízký.

Vždy je dobré mít v systému alespoň nějaký odkládací prostor, i když podle výpočtů žádný nepo-třebujete. Linux používá odkládací prostor poněkud agresivněji, snaží se mít tolik volné fyzické paměti, kolik je jenom možné. Linux navíc odkládá paměťové stránky, které se nepoužívají, i když se fyzická paměť zatím nevyužívá. Tím se totiž zamezí čekání na odložení v případě akutní potře-by, data se odkládají dříve, v době, kdy je disk jinak nevyužitý.

Odkládací prostor lze rozdělit mezi několik disků. To může v některých případech zlepšit výkon systému, jenž v tomto směru závisí na relativních rychlostech disků a jejich přístupových mode-lech. Lze samozřejmě experimentovat s několika variantami, ale mějte vždy na paměti to, že je velmi lehké u takovýchto pokusů pochybit. Neměli byste také věřit tvrzením, že některá z mož-ností je lepší než ostatní, protože to nemusí být vždy pravda.

Vyrovnávací pamě

Čtení z disku je ve srovnání s přístupem k fyzické paměti velmi pomalé. Navíc se v běžném pro-vozu velmi často z disku načítají stejná data několikrát během relativně krátkých časových inter-valů. Například v případě elektronické pošty se nejdříve musí načíst došlá zpráva; když na ni chce-te odpovědět, načte se ten samý dopis do editoru; pak stejná data načte i poštovní program, který je kopíruje do souboru s došlou (případně odeslanou) poštou a podobně. Nebo si zkuste před-stavit, jak často se může zadávat příkaz ls na systému s mnoha uživateli. Jediným načtením infor-mací z disku a jejich uložením do paměti do doby, až je nebude potřeba, lze zrychlit všechny ope-race čtení z disku s výjimkou prvního čtení. Paměť vyhrazená pro tyto účely, tedy pro ukládání z disku načítaných a na disk zapisovaných dat, se nazývá *vyrovnávací paměť*.

Paměť je naneštěstí omezený a navíc vzácný systémový prostředek, takže vyrovnávací paměť nemůže být obvykle dost velká (nemohou v ní být uložena úplně všechna data, která by chtěl někdo používat). Když se vyrovnávací paměť zaplní, data, jež se nepoužívala nejdéle, se zruší a takto uvolněná vyrovnávací paměť se využije na ukládání nových dat.
Vyrovnávací paměť funguje i při zápisu na disk. Jednak proto, že data, která se zapisují na disk, se velmi často brzo opakovaně načítají (například zdrojový kód nejprve uložíme do souboru a pak jej opět načítá překladač), takže je výhodné uložit data zapisovaná na disk i do vyrovnávací pamě-ti. Druhou výhodou je to, že uložením dat do vyrovnávací paměti (aniž by se okamžitě zapsala na disk) se zlepší odezva programu, jenž data zapisuje. Zápis dat na disk pak může probíhat na poza-dí bez toho, že by se tím zpomalovaly ostatní programy.

Diskové vyrovnávací paměti používá většina operačních systémů (i když je možné, že se jim říká nějak jinak). Ne všechny ale pracují podle výše uvedených principů. Některé fungují jako *write-through*: Data se zapisují na disk ihned (ovšem přirozeně zůstanou rovněž uložena ve vyrovná-vací paměti). Je-li zápis odložen na pozdější dobu, označují se tyto vyrovnávací paměti jako *wri-teback*. Tento systém je samozřejmě efektivnější, je ale také o něco více náchylný k chybám. V pří-padě havárie systému, případně výpadku proudu v nevhodném okamžiku či vytažení diskety z disketové mechaniky předtím, než jsou data čekající ve vyrovnávací paměti na uložení zapsána, se změny uložené ve vyrovnávací paměti obvykle ztratí. Navíc by takováto událost mohla zapři-činit chyby v souborovém systému (je-li na disku či disketě vytvořen), jelikož mezi nezapsanými daty mohou být i důležité změny účetních informací samotného souborového systému.

Proto by se nikdy nemělo vypínat napájení počítače dřív, než správně proběhla procedura zasta-vení systému. Rovněž by se neměla vytahovat disketa z mechaniky předtím, než byla odpojena příkazem umount (byla-li předtím připojená), případně

předtím, než program, jenž přistupoval na disketu, signalizuje, že práci s disketovou mechanikou ukončil, a také než přestane svítit kon-trolka mechaniky pružného disku. Příkaz sync vyprázdní vyrovnávací paměť, tedy uloží všechna nezapsaná data na disk. Lze jej použít vždy, když chcete obsah vyrovnávací paměti bezpečně ulo-žit na disk. V tradičních systémech Unix existuje program update, který běží na pozadí a spouští příkaz sync každých 30 sekund. V těchto systémech proto není obvykle potřeba příkaz sync pou-žívat ručně. V systému Linux běží další démon bdflush, jenž dělá něco podobného jako program sync, ale častěji a ne v takovém rozsahu. Navíc se tímto způsobem vyhnete náhlému „zamrznutí“ systému, které může někdy příkaz sync při větším rozsahu vstupně-výstupních operací způsobit.

V systému Linux se program bdflush spouští příkazem update. V případě, že je démon bdflush z jakéhokoliv důvodu neočekávaně ukončen, jádro systému o tom podá varovné hlášení. Obvyk-le ale není důvod se něčeho obávat. Proces bdflush lze spustit ručně (příkazem /sbin/update).

Ve skutečnosti se do vyrovnávací paměti neukládají celé soubory, ale bloky, což jsou nejmenší jednotky při vstupně-výstupních diskových operacích (v Linuxu mají nejčastěji velikost 1 kB). Tímto způsobem se do vyrovnávací paměti ukládají i adresáře, superbloky, další účetní data sou-borového systému a data z disků bez souborových systémů.

O efektivitě vyrovnávací paměti prvotně rozhoduje její velikost. Malá vyrovnávací paměť je téměř zbytečná. Bude v ní uloženo tak málo dat, že se vyrovnávací paměť vždy vyprázdní předtím, než mohou být data opakovaně použita. Kritická velikost záleží na tom, jaké množství dat se z disků čte a na disky zapisuje a jak často se k těmto údajům opakovaně přistupuje. Jediný způsob, jak to zjistit, je experimentovat.

Má-li vyrovnávací paměť pevnou velikost, není výhodné ji mít příliš velkou. To by mohlo kriticky zmenšit dostupnou systémovou paměť a zapříčinit časté odkládání (jež je rovněž pomalé). Aby byla reálná paměť využívána co nejefektivněji, Linux automaticky využívá veškerou volnou paměť RAM jako vyrovnávací paměť. Naopak, operační systém vyrovnávací paměť automaticky zmenšuje, když běžící programy požadují víc fyzické paměti.

O vyrovnávací paměť se v Linuxu nemusíte nijak starat, vše probíhá automaticky. Kromě dodržo-vání korektních postupů vypínání počítače a odpojování médií si jí vůbec nemusíte všímat.

Sledování systému

„Máš dozor na chodbě!“ Spongebob Squarepants (animovaný seriál)

Jedním z nejdůležitějších úkolů správce systému je sledování systému. Jako správce musíte být schopen zjistit, co se v dané chvíli v systému děje. Může to být využívání systémových prostřed-ků, které příkazy se provádějí, kdo je přihlášený. V této kapitole si řekneme o tom, jak se moni-toruje systém, a v některých případech i o řešení problémů, které nastaly.

Při zvyšování výkonu systému je nutno uvažovat o čtyřech hlavních oblastech: základní jednotka, paměť, disky a síť. Umíte-li v systému nalézt kritické místo, ušetříte spoustu času.

Systémové prostředky

Nejdůležitější je sledování výkonu počítače. Nedostatek systémových prostředků může způsobit značné problémy. Mohou je využívat jak uživatelé, tak i služby spojené s činností systému jako elektronická pošta nebo internetové stránky. Když víte, co se v systému děje, snáze můžete roz-hodovat o tom, zda je nutno posílit systém nebo jestli by bylo lepší přesunout některé služby na jiný počítač.

Příkaz top

Základním příkazem pro monitorování systému je příkaz top, který kontinuálně vypisuje aktuální hlášení o využívání systémových prostředků.

```
# top
  12:10:49 up 1 day, 3:47, 7 users, load average: 0.23, 0.19, 0.10 125 processes: 105 sleeping, 2 running, 18 zombie, 0 stopped CPU states: 5.1% user 1.1%
system 0.0% nice 0.0% iowait 93.6% idle Mem: 512716k av,  506176k used,  6540k free,  0k shrd,  21888k buff Swap: 1044216k av,  161672k used,
882544k free  199388k cached

    PID USER PRI NI SIZE RSS SHARE STAT %CPU %MEM TIME CPU COMMAND
  2330 root 15 0 161M 70M 2132 S 4.9 14.0 1000m 0 X
  2605 weeksa 15 0 8240 6340 3804 S 0.3  1.2 1:12 0 kdeinit
```

3413 weeksa 15 0 6668 5324 3216 R 0.3 1.0 0:20 0 kdeinit 18734 root 15 0 1192 1192 868 R 0.3 0.2 0:00 0 top
1619 root 15 0 776 608 504 S 0.1 0.1 0:53 0 dhclient
1 root 15 0 480 448 424 S 0.0 0.0 0:03 0 init
2 root 15 0 0 0 0 SW 0.0 0.0 0:00 0 keventd 3 root 15 0 0 0 0 SW 0.0 0.0 0:00 0 kapmd
4 root 35 19 0 0 0 SWN 0.0 0.0 0:00 0 ksoftirqd_CPU0
9 root 25 0 0 0 0 SW 0.0 0.0 0:00 0 bdflush
5 root 15 0 0 0 0 SW 0.0 0.0 0:00 0 kswapd
10 root 15 0 0 0 0 SW 0.0 0.0 0:00 0 kupdated
11 root 25 0 0 0 0 SW 0.0 0.0 0:00 0 mdrecoveryd
15 root 15 0 0 0 0 SW 0.0 0.0 0:01 0 kjournald
81 root 25 0 0 0 0 SW 0.0 0.0 0:00 0 khubd
1188 root 15 0 0 0 0 SW 0.0 0.0 0:00 0 kjournald
1675 root 15 0 604 572 520 S 0.0 0.1 0:00 0 syslogd
1679 root 15 0 428 376 372 S 0.0 0.0 0:00 0 klogd

1813 root 25 0 752 528 524 S 0.0 0.1 0:00 0 sshd1828 root 25 0 704 548 544 S 0.0 0.1 0:00 0 xinetd

<zkráceno>

V horní části hlášení jsou informace o systémovém čase, době bezporuchového chodu, využívání základní jednotky, využívání fyzické a odkládací paměti a o počtu procesů. Pod ní je seznam procesů setříděných podle doby využívání základní jednotky počítače.

Výstup příkazu top můžete v průběhu činnosti modifikovat. Když stisknete klávesu i (jako idle), program přestane vypisovat nečinné procesy. Opětovným stisknutím i výpis obnovíte. Stisknutím klávesy M setřídíte procesy podle toho, jak využívají paměť, klávesou S podle délky běhu proce-su a klávesou P se vrátíte k původnímu třídění podle doby využívání základní jednotky počítače.

Kromě volby způsobu zobrazení můžete procesy příkazem top také modifikovat. Pomocí u může-te prohlížet procesy, které náleží určitému uživateli, pomocí k můžete procesy rušit a pomocí r můžete měnit parametry procesů.

Podrobnější informace o procesech naleznete v souborovém systému /proc, který obsahuje řadu podadresářů s číselnými jmény. Vztahují se k ID běžících procesů a naleznete v nich soubory s informacemi o těchto procesech.

TYTO SOUBORY NESMÍTE V ŽÁDNÉM PŘÍPADĚ ZMĚNIT – MOHLO BY DOJÍT K POŠKOZENÍ SYSTÉMU!

Příkaz iostat

Příkaz iostat vypisuje průběžné zatížení základní jednotky a informace o V/V. Je to vynikající pří-kaz pro sledování využití disků.

iostat Linux 2.4.20-24.9 (myhost) 12/23/2003

avg-cpu: %user %nice %sys %idle
62.09 0.32 2.97 34.62

Device: tps Blk_read/s Blk_wrtn/s Blk_read Blk_wrtn dev3-0 2.22 15.20 47.16 1546846 4799520

V jádru 2.4 a 2.6 vypisuje tento příkaz hlavní a vedlejší číslo zařízení. V tomto případě je to /dev/hda. V příkazu iostat zadejte volbu -x.

iostat -x Linux 2.4.20-24.9 (myhost) 12/23/2003

avg-cpu: %user %nice %sys %idle
62.01 0.32 2.97 34.71

Device: rrqm/s wrqm/s r/s w/s rsec/s wsec/s rkB/s wkB/s avgrq-sz avgqu-sz await svctm %util
/dev/hdc 0.00 0.00 .00 0.00 0.00 0.00 0.00 0.00 0.00 2.35 0.0 0 0.00 14.71
/dev/hda 1.13 4.50 .81 1.39 15.18 47.14 7.59 23.57 28.24 1.99 63.76 70.48 15.56
/dev/hda1 1.08 3.98 .73 1.27 14.49 42.05 7.25 21.02 28.22 0.44 21.82 4.97 1.00
/dev/hda2 0.00 0.51 .07 0.12 0.55 5.07 0.27 2.54 30.35 0.97

52.67 61.73 2.99

/dev/hda3 0.05 0.01 .02 0.00 0.14 0.02 0.07 0.01 8.51 0.00

12.55 2.95 0.01

Význam jednotlivých sloupců je popsán v manuálových stránkách iostat.

Příkaz ps

Seznam běžících procesů. Příkaz má mnoho různých voleb.Příkazem obvykle vypisujeme všechny běžící procesy. Použijte volbu ps -ef. (Celkový výpis jevelmi dlouhý, je tedy uvedena jenom část.)

UID PID PPID C STIME TTY TIME CMD root 1 0 0 Dec22 ? 00:00:03 init root 2 1 0 Dec22 ? 00:00:00 [keventd] root 3 1 0 Dec22 ? 00:00:00 [kapmd] root 4 1 0 Dec22 ? 00:00:00 [ksoftirqd_CPU0] root 9 1 0 Dec22 ? 00:00:00 [bdflush] root 5 1 0 Dec22 ? 00:00:00 [kswapd] root 6 1 0 Dec22 ? 00:00:00 [kscand/DMA] root 7 1 0 Dec22 ? 00:01:28 [kscand/Normal] root 8 1 0 Dec22 ? 00:00:00 [kscand/HighMem] root 10 1 0 Dec22 ? 00:00:00 [kupdated] root 11 1 0 Dec22 ? 00:00:00 [mdrecoveryd] root 15 1 0 Dec22 ? 00:00:01 [kjournald] root 81 1 0 Dec22 ? 00:00:00 [khubd] root 1188 1 0 Dec22 ? 00:00:00 [kjournald] root 1675 1 0 Dec22 ? 00:00:00 syslogd -m 0 root 1679 1 0 Dec22 ? 00:00:00 klogd -x rpc 1707 1 0 Dec22 ? 00:00:00 portmap root 1813 1 0 Dec22 ? 00:00:00 /usr/sbin/sshd ntp 1847 1 0 Dec22 ? 00:00:00 ntpd -U ntp root 1930 1 0 Dec22 ? 00:00:00 rpc.rquotad root 1934 1 0 Dec22 ? 00:00:00 [nfsd] root 1942 1 0 Dec22 ? 00:00:00 [lockd] root 1943 1 0 Dec22 ? 00:00:00 [rpciod] root 1949 1 0 Dec22 ? 00:00:00 rpc.mountd root 1961 1 0 Dec22 ? 00:00:00 /usr/sbin/vsftpd
/etc/vsftpd/vsftpd.conf

root 2057 1 0 Dec22 ? 00:00:00 /usr/bin/spamd -d -c -a

bin 2076 1 0 Dec22 ? 00:00:00 /usr/sbin/cannaserver -syslog -u bin root 2087 1 0 Dec22 ? 00:00:00 crond <zkráceno>

V prvním sloupci je uveden vlastník procesu, v druhém sloupci je ID procesu. Ve třetím sloupci je ID rodiče, což je proces, který daný proces vytvořil nebo spustil. Ve čtvrtém sloupci je využití základní jednotky v procentech. Pátý sloupec obsahuje dobu spuštění procesu, případně i datum, běží-li proces dlouho. V šestém sloupci je tty, které náleží k danému procesu, pokud existuje. Sedmý sloupec obsahuje souhrnný čas využití základní jednotky (celkový čas základní jednotky spotřebovaný daným procesem). V sedmém sloupci je příkaz samotný.

S využitím těchto informací je zřejmé, co se v systému děje. Bezprizorní procesy a procesy, které způsobují problémy, můžete ukončit.

Příkaz vmstat

Příkaz vmstat poskytuje hlášení, které obsahuje statistiku systémových procesů, paměti, odkládá-ní, V/V a základní jednotky. Statistika se vytváří z dat od posledního zadání tohoto příkazu do pří-tomnosti. Nebyl-li příkaz ještě použit, berou se data od spuštění systému.

vmstat

procs			memory				swap		io		system		cpu	
r	b	w	swpd	free	buff	cache	si	so	bi	bo	in	cs	us	sy id
0	0	0	181604	17000	26296	201120	0	2	8	24	149	9	61	3 36

Následující popis polí je z manuálových stránek vmstat: Procs
r: Počet procesů čekajících na spuštění.

b: Počet procesů v nepřerušitelném spánku.

w: Počet odložených běžících procesů. Linux nikdy neodkládá procesy bezdůvodně.

Memory swpd: Velikost použité virtuální paměti (kB). free: Velikost nevyužité paměti (kB). buff: Velikost paměti využívané pro buffery (kB). cache: Velikost paměti využívané pro cache (kB).

Swap si: Velikost paměti přenášené z disku (kB/s). so: Velikost paměti přenášené na disk (kB/s).

IO bi: Bloky posílané na blokové zařízení (blocks/s). bo: Bloky čtené z blokového zařízení (blocks/s).

System in: Počet přerušení za vteřinu včetně přerušení od hodin. cs: Počet souvisejících přepnutí za vteřinu.
CPU
Doba využití základní jednotky v procentech.
us: uživatel

sy: systém
id: běh naprázdno

Příkaz lsof

Příkaz lsof vypíše seznam všech právě používaných souborů. Vzhledem k tomu, že Linux pova-žuje za soubor všechno, může být seznam velmi dlouhý. Při diagnostických problémech je však tento příkaz velmi užitečný. Za příklad může posloužit situace, kdy chcete odpojit souborový systém, avšak nelze to provést kvůli tomu, že jej někdo používá. Pomocí tohoto příkazu a pomo-cí příkazu grep zjistíte, kdo tento soubor používá.

Anebo předpokládejte, že chcete vidět všechny soubory používané určitým procesem. V takovém případě stačí zadat příkaz lsof -p -processid-.

Kde najdete další nástroje

Více se o řádkových nástrojích dočtete v referenční příručce, kterou sepsal Chris Karakas a najde-te ji na adrese GNU/Linux Command-Line Tools Summary (*http://www.karakas-online.de /gnu-linux-tools-summary/*). Je vhodným zdrojem pro vyhledání dalších nástrojů a návodů k jejich používání.

Souborový systém

Neustále se dočítáme, jak je diskový prostor levný, avšak převážná část uživatelů s tím nemůže souhlasit. Většina z nás se neustále potýká s nedostatkem místa na disku. Potřebovali bychom sle-dovat a řídit jeho využívání.

Příkaz df

Příkaz df je nejjednodušším nástrojem, kterým můžeme sledovat využití disku. Zadáte pouze df a vypíše se vám využití všech připojených disků v blocích o velikosti 1 kB.

user@server:~> df

Filesystem	1K-blocks		Used	Available	Use%	Mounted on
/dev/hda3	5242904	759692	4483212	15%	/	
tmpfs	127876	8	127868	1%	/dev/shm	
/dev/hda1	127351	33047	87729	28%	/boot	
/dev/hda9	10485816	33508	10452308	1%	/home	
/dev/hda8	5242904	932468	4310436	18%	/srv	
/dev/hda7	3145816	32964	3112852	2%	/tmp	
/dev/hda5	5160416	474336	4423928	10%	/usr	
/dev/hda6	3145816	412132	2733684	14%	/var	

Pomocí volby -h dostanete výstup v „čitelnější“ podobě. Velikosti budou uvedeny v kilobajtech, megabajtech nebo gigabajtech v závislosti na velikosti souborového systému. Velikost bloku zadá-te volbou -B.

Kromě využití prostoru na disku si pomocí volby -i můžete také zobrazit počet použitých a vol-ných čísel inod.
user@server:~> df -i Filesystem Inodes IUsed IFree IUse% Mounted on /dev/hda3 0 0 0 -/ tmpfs 31969 5 31964 1% /dev/shm /dev/hda1 32912 47 32865 1% /boot /dev/hda9 0 0 0 -/home /dev/hda8 0 0 0 -/srv /dev/hda7 0 0 0 -/tmp /dev/hda5 656640 26651 629989 5% /usr /dev/hda6 0 0 0 -/var

Příkaz du

Teď už víte, kolik místa máte na souborových systémech zabráno, jak ale zjistíte, kde konkrétně data jsou? Například pomocí příkazu du. Pokud neuvedete jméno souboru, příkaz du bude fun-govat rekurzivně. Například:

user@server:~> du file.txt 1300 file.txt

Nebo jako u příkazu df můžete použít volbu -h a výstup se vypíše v čitelnější podobě.

user@server:~> du -h file.txt 1.3M file.txt

Pokud neuvedete jméno souboru, příkaz du bude fungovat rekurzivně.

user@server:~> du -h /usr/local 4.0K /usr/local/games 16K /usr/local/include/nessus/net
180K /usr/local/include/nessus 208K /usr/local/include 62M /usr/local/lib/nessus/plugins/.desc
97M /usr/local/lib/nessus/plugins 164K /usr/local/lib/nessus/plugins_factory <zkráceno>

Chcete-li vypsat pouze souhrnné číslo pro daný adresář, provedete to pomocí volby -s.

user@server:~> du -hs /usr/local 210M /usr/local

Quotas

Informace o diskových kvótách naleznete v The Quota HOWTO na adrese *http://www.tldp.org/HOWTO/Quota.html.*

Monitorování uživatelů

To, že jste paranoik, neznamená, že vás nedostanou... Neznámý autor

Čas od času vás začne zajímat, co provádějí uživatelé v systému. Zjistíte třeba, že se nadměrně využívá vyrovnávací paměť RAM nebo základní jednotka. Určitě se budete chtít podívat, kdo je v systému, co provozuje a jaké k tomu využívá prostředky.

Příkaz who

Nejjednodušší způsob zjištění, kdo je v systému, je zadat who nebo w. Tímto příkazem si vypí-šete, kdo je přihlášen do systému a na kterém portu, resp. terminálu.

user@server:~> who bjones pts/0 May 23 09:33 wally pts/3 May 20 11:35 aweeks pts/1 May 22 11:03 aweeks pts/2 May 23 15:04

Ještě jednou příkaz ps!

V předchozí kapitole jsme viděli, že uživatel aweeks je přihlášený jak na pts/1, tak i na pts/2, avšak co když chceme vědět, co na nich dělá? Zadáme ps -u aweeks a dostaneme výstup:

user@server:~> ps -u aweeks

20876 pts/1 00:00:00 bash 20904 pts/2 00:00:00 bash 20951 pts/2 00:00:00 ssh 21012 pts/1 00:00:00 ps

Vidíme, že tento uživatel provozuje ps a ssh.

Příkaz w

Ještě jednodušší než příkazy who a ps -u je příkaz w, který vypíše, nejen kdo je přihlášen do systému, ale také jaký příkaz provádí.

user@server:~> w

aweeks	:0	09:32	?xdm? 30:09		0.02s -:0
aweeks	pts/0	09:33	5:49m	0.00s	0.82s kdeinit: kded
aweeks	pts/2	09:35	8.00s	0.55s	0.36s vi sag-0.9.sgml
aweeks	pts/1	15:03	59.00s	0.03s	0.03s /bin/bash

Vidíme, že mám spuštěný program kde, pracuji na tomto dokumentu :-) a jsem přihlášený ještě na jednom terminálu, který běží naprázdno s vypsaným promptem bash.

Spouštění a zastavování systému

Start me up Ah... you've got to... you've got to Never, never never stop Start it up Ah... start it up, never, never, never You make a grown man cry, you make a grown man cry
(Rolling Stones)

Tato kapitola popisuje, co se děje po tom, co je systém Linux spuštěn, a jak jej správně zastavit. Při nedodržení správných postupů může dojít k poškození nebo ztrátě souborů.

Zavádění a ukončení práce systému – přehled

Zapnutí počítače a následné zavedení operačního systému se označuje jako *bootování*. Tento ter-mín původně vznikl z anglické fráze „pull yourself up by your own bootstraps“, tedy vytáhnout sebe sama za jazyk vlastních bot, což obrazně vystihuje proces, který probíhá při zapnutí počíta-če – nicméně realita je podstatně méně zábavná.

V průběhu zavádění počítač nejdříve nahraje krátký kód, takzvaný *zavaděč*, který pak zavede a spustí samotný operační systém. Zavaděč je obvykle uložen na předem určeném místě pevné-ho disku nebo diskety. Důvodem pro rozdělení procedury zavádění systému do dvou kroků je to, že samotný operační systém je velký a složitý, kdežto samotný zavaděč je velmi krátký (má něko-lik stovek bajtů). Tím se zamezí nežádoucímu komplikování firmwaru.

Různé typy počítačů provádí úvodní sekvence zavádění systému různě. Pokud jde o počítače třídy PC, ty (přesněji jejich systém BIOS) načítají první sektor pevného disku nebo diskety, kterému se říká *zaváděcí sektor*. Zavaděč je uložen v tomto prvním – zaváděcím sektoru. Zavaděč pak načítá operační systém z jiného místa na disku, případně i z nějakého jiného média.

Poté co se operační systém Linux zavede, inicializují se technické prostředky počítače a ovladače jednotlivých zařízení. Pak se spustí proces init, který spouští další procesy umožňující uživatelům přihlásit se do systému a pracovat v něm. O podrobnostech této části zaváděcího procesu se pojednává dále.
Aby bylo možné systém Linux zastavit, je nejdříve nutné požádat všechny procesy, aby ukončily činnost (zavřely všechny otevřené soubory, popřípadě zařídily další důležité věci – obrazně řeče-no „uklidily“ po sobě). Potom se odpojí souborové systémy, odkládací prostory a nakonec se na konzole objeví zpráva, že lze počítač vypnout. Jestli se nedodrží správný postup, mohou se stát (a obvykle se stanou) dost nepříjemné věci. Nejzávažnějším problémem je v tomto případě nevy-prázdněná vyrovnávací disková paměť souborového systému. Všechna data ve vyrovnávací pamě-ti se totiž ztratí, takže souborový systém na disku bude nekonzistentní a pravděpodobně nepou-žitelný.

Zavádění podrobněji

Systém Linux lze zavést buď z disket, z pevného disku nebo z CD. Příslušná kapitola Průvodce instalací uvádí návod, jak systém instalovat všemi výše uvedenými způsoby. Když počítač PC star-tuje, systém BIOS provádí různé testy a kontroluje, zda je vše v pořádku. Až pak zahájí skutečné zavádění operačního systému. Vybere zaváděcí diskovou jednotku. Typicky první disketovou mechaniku (je-li v mechanice zasunuta disketa), jinak první pevný disk, pokud je v počítači nain-stalován. Pořadí prohledávání lze konfigurovat. Poté se načte první sektor tohoto disku, kterému se říká *zaváděcí sektor*; u pevných disků se označuje jako *hlavní zaváděcí sektor*, protože na pev-ném disku může být několik diskových oblastí, každá se svým vlastním zaváděcím sektorem.

Zaváděcí sektor obsahuje krátký program (dostatečně krátký na to, aby se vešel do jednoho bloku), jehož úkolem je načíst z disku operační systém a spustit jej. Při zavádění systému z dis-kety obsahuje její zaváděcí sektor kód, který načte jenom prvních několik stovek bloků (v závis-losti na aktuální velikosti jádra) do předem určeného místa v paměti. Na linuxové zaváděcí dis-ketě totiž nebývá vytvořen souborový systém, je na ní uloženo jenom jádro systému v několika po sobě jdoucích sektorech, což proces zavádění systému značně zjednodušuje. Systém lze zavést rovněž z diskety se souborovým systémem, a to pomocí zavaděče systému Linux (anglicky LInux LOader, zkráceně LILO).

Když se systém zavádí z pevného disku, kód obsažený v zaváděcím sektoru disku si prohlédne tabulku diskových oblastí, která je v tomto sektoru také uložená. Pak zaváděcí kód identifikuje aktivní diskovou oblast (oblast, která je označená jako zaváděcí), načte zaváděcí sektor této dis-kové oblasti a spustí kód v něm uložený. Kód v zaváděcím sektoru diskové oblasti pak dělá v pod-statě to samé, co kód obsažený v zaváděcím sektoru diskety. Načte jádro systému ze své diskové oblasti a spustí jej. Avšak v detailech se tyto procedury poněkud liší, protože obecně není výhod-né mít zvláštní diskovou oblast čistě pro obraz jádra systému, proto kód v zaváděcím sektoru nemůže jednoduše sekvenčně číst data z disku jako při zavádění systému z diskety. Existuje něko-lik způsobů řešení tohoto problému. Nejběžnější možností je použít zavaděč operačního systému Linux LILO. (Další detaily tohoto postupu nejsou v této chvíli podstatné. Více informací najdete v dokumentaci zavaděče LILO, která je v tomto směru dokonalejší.)

Když se zavádí operační systém pomocí zavaděče LILO, pokračuje se načtením a spuštěním impli-citního jádra. Je také možné nakonfigurovat LILO tak, aby zavedl některý z více obrazů jádra systé-mu, nebo i jiný operační systém než Linux. Uživatel systému si tak při zavádění může vybrat, které jádro, případně operační systém, se implicitně zavede při spuštění počítače. Zavaděč LILO lze nakonfigurovat i tak, že při zmáčknutí kláves Alt, Shift nebo Control v okamžiku zavádění systé-mu (tedy při spouštění LILO) se nebude zavádět systém ihned. Místo toho se zavaděč zeptá, který operační systém se bude zavádět. LILO lze nastavit i tak, že se bude při zavádění systému ptát na požadovaný systém, ale s volitelným časovým prodlením, po kterém se zavede implicitně určené jádro.
Zavaděč LILO rovněž umožňuje předat jádru systému řádkové parametry, které se zadávají za jménem jádra nebo operačního systému, jenž se zavádí. Přehled možných voleb naleznete na adrese
http://www.tldp.org/HOWTO/BootPrompt-HOWTO.html.

Zavádění systému z diskety i z pevného disku má své výhody. Obecně je zavádění z disku pří-jemnější, uživatel je ušetřen

zbytečných nepříjemností v případě, že v disketách „zcela náhodou“ zavládne nepořádek. Kromě toho je zavádění z disku rychlejší. Většina linuxových distribucí vytvoří zavaděč v průběhu instalačního procesu.

Jakmile se jádro systému načte do paměti (ať už to znamená cokoliv) a dojde k jeho skutečnému spuštění, stanou se přibližně následující věci:

Jádro Linuxu se instaluje v komprimovaném tvaru, takže se nejprve samo dekomprimuje. Stará se o to krátký program, jenž je obsažen v začátku obrazu jádra.
Rozezná-li systém kartu Super VGA, která má nějaké zvláštní textové režimy (jako napří-klad 100 sloupců na 40 řádků), zeptá se vás, který z režimů budete používat. V průběhu kompilace jádra systému lze videorežim nastavit – pak se systém při startu nedotazuje. Stej-ného efektu lze dosáhnout konfigurací zavaděče systému LILO, GRUB nebo příkazem rdev.
V dalším kroku jádro zkontroluje, jaké další hardwarové komponenty jsou k dispozici (pevné disky, diskety, síťové adaptéry a podobně), a odpovídajícím způsobem nastaví některé ze svých ovladačů zařízení. V průběhu této „inventury“ vypisuje jádro zprávy

> o tom, která zařízení byla nalezena. Například při zavádění systému, jejž používá autor, se vypisují tato hlášení:
>
> LILO boot: Loading linux. Console: colour EGA+ 80x25, 8 virtual consoles Serial driver version 3.94 with no serial options enabled tty00 at 0x03f8 (irq = 4) is a 16450 tty01 at 0x02f8 (irq = 3) is a 16450 lp_init: lp1 exists (0), using polling driver Memory: 7332k/8192k available (300k kernel code, 384k reserved, 176k data) Floppy drive(s): fd0 is 1.44M, fd1 is 1.2M Loopback device init Warning WD8013 board not found at i/o = 280. Math coprocessor using irq13 error reporting. Partition check: hda: hda1 hda2 hda3 VFS: Mounted root (ext filesystem). Linux version 0.99.pl9-1 (root@haven) 05/01/93 14:12:20

Přesný formát výstupů se na různých systémech liší, a to v závislosti na hardwarové kon-figuraci výpočetního systému, verzi operačního systému a jeho konkrétním nastavení.
Potom se jádro Linuxu pokusí připojit kořenový svazek. Přípojné místo lze nastavit při kompilaci jádra, později pak příkazem rdev, případně zavaděčem. Typ souborového syste-mu je detekován automaticky. Jestli připojení souborového systému selže (například proto, že jste při kompilaci jádra systému zapomněli uvést odpovídající ovladač souborového systému), jádro zpanikaří a systém se v tomto kroku zastaví (beztak mu nic jiného ani nezbývá).

Kořenový svazek se obvykle připojuje pouze pro čtení (to lze nastavit stejným způsobem jako místo připojení). Díky tomu se může provést kontrola souborového systému při jeho připojování. Není vhodné prověřovat systém souborů, který je připojen pro čtení i zápis.

Poté spustí jádro systému na pozadí program init, jenž se nachází v adresáři /sbin/init. Proces init bude vždy procesem číslo 1 a jeho úkolem jsou různé operace spojené se star-tem systému. Přesný postup toho, co jádro systému v tomto kroku dělá, závisí na tom, jak je konfigurováno; viz kapitolu „Proces init“. Jádro systému v této fázi přinejmenším spustí na pozadí některé nezbytné démony.
Proces init pak přepne do víceuživatelského režimu a spustí procesy getty pro virtuální konzoly a sériové linky. Proces getty je program, který umožňuje uživatelům přihlásit se prostřednictvím virtuální konzoly nebo sériových terminálů do systému. Proces init může rovněž spouštět některé další programy, a to podle toho, jak je konfigurován.
Poté je zavádění systému ukončeno a systém normálně běží.

Poznámka o zavaděčích

Informace o LILO naleznete na adrese *http://www.tldp.org/HOWTO/LILO.html.* Informace o GRUB naleznete na adrese *http://www.gnu.org/software/grub/grub.html.*

Podrobněji o zastavení systému

I při zastavování operačního systému Linux je důležité dodržovat správný postup. Pokud se správ-ná procedura zastavení systému nedodrží, budou souborové systémy pravděpodobně poškozeny a obsah jednotlivých souborů může být promíchán. To proto, že operační systém využívá disko-vou vyrovnávací paměť, jež nezapisuje změny na disk ihned, ale v určitých časových intervalech. To sice významně zvyšuje výkon systému, ale následkem toho je, že pokud se z ničeho nic vypne napájení ve chvíli, kdy vyrovnávací paměť obsahuje množství nezapsaných změn, mohou být data na disku nekonzistentní a souborový systém zcela nefunkční, protože se na disk zapsala jenom některá změněná data.

Dalším z argumentů proti tvrdému vypnutí počítače je to, že v systému ve víceuživatelském režimu může běžet hodně programů na pozadí. Vypnutí ze sítě by pak mohlo mít katastrofální důsledky. Tím, že se při zastavení systému dodržuje korektní postup, se zajistí, že všechny procesy běžící na pozadí svá data včas uloží.

Běh systému Linux se správně ukončí příkazem shutdown. Zadává se obvykle jedním ze dvou způsobů. Když jste přihlášeni v systému jako jediný uživatel, je potřeba před zadáním příkazu shutdown ukončit všechny běžící programy, odhlásit se ze všech virtuálních konzol a přihlásit se na jednu z nich jako superuživatel. Pokud už tak jste přihlášení, je rozumné přepnout se buď do kořeno-vého adresáře nebo do svého domovského adresáře, aby se předešlo problémům při odpojení

souborových systémů. Pak můžete zadat příkaz shutdown -h now. (Místo parametru now může-te zadat symbol plus a nějaké číslo, pak se zastavení systému zpozdí o zadaný počet minut – ve většině případů totiž nejste jediný uživatel systému.)

Druhou alternativou – je-li v systému přihlášeno více uživatelů – je použití příkazu shutdown -h +*time message*, kde *time* je čas v minutách zbývající do zastavení systému a *message* je stručné sdělení důvodu tohoto opatření.

```
# shutdown -h +10 'Bude instalován nový disk. Systém by > měl být opět spuštěn za tři hodiny.'
#
```

Takto můžete každého uživatele varovat, že systém bude za deset minut zastaven a že bude lepší se odpojit, než ztratit neuložená data. Varování se zobrazí na každém terminálu, na kterém je někdo přihlášený, včetně všech terminálů xterm systému X Window:

```
Broadcast message from root (ttyp0) Wed Aug 2 01:03:25 1995...
Bude instalován nový disk. Systém by měl být opět spuštěn za tři hodiny.
The system is going DOWN for system halt in 10 minutes !!
```

Varování se automaticky opakuje několik minut před zastavením systému v kratších a kratších

intervalech, až stanovený čas vyprší.Když se pak po určeném časovém prodlení rozjede procedura skutečného zastavení systému,odpojí se nejdříve všechny souborové systémy (kromě kořenového), uživatelské procesy (je-liněkdo stále přihlášen) se ukončí, běžící démoni se zastaví, všechny připojené svazky se odpojí,obrazně řečeno – všechno ustane. Poté proces init vypíše zprávu, že lze počítač vypnout. Až paklze sáhnout na síťový vypínač.

V některých případech (ovšem zřídkakdy u správně nakonfigurovaného systému) není možnézastavit systém korektně. Například když jádro systému zpanikaří, havaruje a nefunguje správně,může být zcela nemožné zadat jakýkoliv další příkaz. Pochopitelně, v takovéto situaci je správnézastavení systému poněkud obtížné. Nezbývá než doufat, že se neuložené soubory až tak mocnepoškodí a vypnout napájení. Když nejsou potíže až tak vážné (řekněme, že někdo jenom seke-rou rozsekl klávesnici vašeho terminálu) a jádro systému i program update stále normálně běží,je obvykle dobré pár minut počkat (dát tím programu update šanci vyprázdnit vyrovnávací pamě-ti) a potom jednoduše vypnout proud.

Někteří uživatelé rádi používají při zastavení systému příkaz sync zadaný třikrát po sobě, pakpočkají, než se ukončí diskové vstupně-výstupní operace, a vypnou napájení. Neběží-li žádné pro-gramy, je tento postup téměř ekvivalentní zadání příkazu shutdown s tím rozdílem, že se neod-pojí žádný ze souborových systémů, což ale může způsobit problémy s nastavením příznakuodpojení ukončení u souborových systémů ext2fs. Proto se metoda trojího zadání příkazu sync*nedoporučuje*.

Pokud vás zajímá, *proč* zrovna třikrát – v raných dobách Unixu se všechny příkazy musely naťu-kat zvlášť, takže než to člověk udělal potřetí, bylo obvykle dost času na to, aby se ukončila větši-na diskových vstupně-výstupních operací.

Znovuzavedení systému

Pod znovuzavedením operačního systému se rozumí jeho opakované zavedení, tedy jeho správ-né zastavení, vypnutí a opětovné zapnutí napájení počítače. Jednodušší cestou je žádost progra-mu shutdown o znovuzavedení systému (místo jeho pouhého zastavení), to použitím parametru -r, tedy například zadáním příkazu shutdown -r now.

Většina systémů Linux vykonává příkaz shutdown -r now i v případě, že se na systémové klá-vesnici současně zmáčknou klávesy ctrl+alt+del. Tato trojkombinace obvykle vyvolá znovuzave-dení operačního systému. Reakci na stisk kláves ctrl+alt+del lze nastavit, a na víceuživatelském počítači by například bylo lepší před znovuzavedením systému povolit určité časové prodlení. Naopak systémy, které jsou fyzicky přístupné komukoliv, by bylo lepší nastavit tak, aby se při zmáčknutí kombinace kláves ctrl-alt-del nedělo vůbec nic.

Jednouživatelský režim

Příkaz shutdown lze použít k přepnutí systému do jednouživatelského režimu. V něm se do systé-mu nemůže přihlásit nikdo jiný než superuživatel, jenž může používat konzolu systému. Jednou-živatelský mód lze s výhodou využít při plnění některých úkolů spojených se správou systému, které nelze dělat, pokud systém běží v normálním (víceuživatelském) režimu.

Záchranné zaváděcí diskety

Občas se stává, že není možné při zapnutí počítače zavést systém z pevného disku. Například tím, že uděláte chybu při nastavování parametrů zavaděče systému LILO, můžete zavinit, že systém Linux nebude možné zavést. V těchto situacích by přišel vhod nějaký jiný způsob zavedení systé-mu, jenž by fungoval vždy (když samozřejmě funguje hardware). Pro počítače PC je takovou alter-nativou zavádění systému z diskety.

Většina distribucí operačního systému Linux umožňuje vytvořit takzvanou *záchrannou zaváděcí disketu* již při instalaci systému. Doporučujeme to udělat. Avšak některé takovéto záchranné dis-kety obsahují pouze jádro systému a předpokládá se, že při

odstraňování vzniklých problémů budete používat programy uložené na instalačních médiích distribuce systému. Někdy ale tyto pro-gramy nestačí. Například v případě, že budete muset obnovit některé soubory ze záloh, jež jste dělali programem, který není na instalačních discích distribuce systému. Proto by si správce měl vytvořit vlastní zaváděcí diskety, přizpůsobené konkrétním potřebám. Pokyny, jak na to, jsou obsaženy v příručce *Bootdisk HOWTO* Grahama Chapmana, viz *http://www.tldp.org/HOWTO/-Bootdisk-HOWTO/index.html.* Musíte mít přirozeně neustále na paměti, abyste měli záchranné zaváděcí diskety stále aktuální.

Disketovou mechaniku, na níž je připojen kořenový souborový systém, nelze použít k ničemu jinému. To může být nevýhodné v případě, že máte jen jedinou mechaniku. Pokud ale máte dosta-tek paměti, můžete vytvořit disketu tak, aby se kořenový souborový systém vytvořil v ramdisku. Pak budete moci disketovou mechaniku využít k libovolným dalším činnostem. Disketovou mechaniku, která se využívá pro připojení superuživatelské zaváděcí diskety, nebudete moci pou-žít k ničemu jinému. (Je proto nutné zvlášť nakonfigurovat jádro systému na zaváděcí disketě.) Když se podaří načíst superuživatelskou zaváděcí disketu na ramdisk, lze disketovou mechaniku využít pro připojení jiných disket.

Proces init

Uuno on numero yksi (slogan z finských filmů)

Tato kapitola popisuje proces init, jenž je vždy prvním uživatelským procesem, který spouští jádro systému. Proces init má mnoho důležitých povinností, spouští například program getty: umožňu-jící uživatelům přihlásit se do systému, implementuje úrovně běhu systému, stará se o osiřelé pro-cesy a podobně. V této kapitole bude vysvětleno, jak se proces init konfiguruje a jak se zavádí různé úrovně běhu systému.

Proces init přichází první

Proces init je jedním z programů, jež jsou sice absolutně nezbytné k tomu, aby operační systém Linux fungoval, ale kterým obvykle nemusíte věnovat přílišnou pozornost. Součástí každé dobré distribuce systému je i předem nastavená konfigurace procesu init, která vyhovuje většině syste-mů. Správce pak už obvykle nemusí kolem procesu init nic dělat. O proces init se většinou sta-ráte, jenom pokud připojujete nové sériové terminály, modemy pro příchozí volání (takzvané *dial -in* modemy, nikoliv tedy modemy, pomocí kterých se budete připojovat do jiných systémů, to jsou *dial-out* modemy) a když potřebujete změnit implicitní úroveň běhu systému.

Když se zavede jádro systému (načte se do paměti, spustí se, inicializuje ovladače zařízení, dato-vé struktury a podobně), ukončí svou roli v proceduře zavádění operačního systému tím, že spus-tí první program uživatelské úrovně – proces init. Takže proces init je vždy prvním spuštěným procesem, má tedy vždy číslo procesu 1.

Jádro systému hledá z historických důvodů proces init na několika místech. Jeho správné umís-tění v systému Linux nicméně je /sbin/init. Nenajde-li jádro proces init, pokusí se spustit pro-gram /bin/sh, a pokud neuspěje, skončí neúspěšně i celá procedura startu systému.

Když proces init nastartuje, dokončí proces zavedení systému tím, že se provede několik admi-nistrativních úkolů, například kontrola souborových systémů, úklid v adresáři /tmp, start různých služeb a spuštění procesu getty pro každý terminál nebo virtuální konzolu, prostřednictvím nichž se uživatelé mohou přihlašovat do systému. (Viz kapitolu „Přihlašování a odhlašování“.)

Po správném zavedení systému proces init po každém odhlášení uživatele restartuje procesy getty pro příslušný terminál. Umožní tím další přihlášení jiných uživatelů. Proces init si také osvojuje všechny osiřelé procesy. Když některý z procesů spustí další proces (svého potomka) a později ukončí svou činnost dřív než potomek, vzniká sirotek, který se ihned stává potomkem procesu init. Adopce sirotků má význam především z různých technických důvodů, je ale dobré o ní vědět, protože je pak snadnější pochopit význam položek seznamu procesů a grafů hierarchické-ho stromu běžících procesů. Program init existuje v několika variantách. Většina linuxových distri-bucí používá sysvinit (napsal jej Miquel van Smoorenburg), založený na programu init ze Systé-mu V. BSD Unix má jiný init. Základní rozdíl je v úrovních běhu: Systém V je má, BSD je nemá (přinejmenším v klasické verzi). Tento rozdíl však není podstatný, budeme se zabývat pouze pro-gramem sysvinit.

Konfigurace procesu init pro spouštění programu getty –

soubor /etc/inittab

Když proces init startuje, načítá konfigurační soubor /etc/inittab. Když pak systém Linux běží, proces init tento konfigurační soubor opakovaně načítá pokaždé, když přijme signál HUP (kill -HUP 1); tato vlastnost umožňuje měnit konfiguraci programu init a zajistit, aby se takováto změna projevila i bez toho, že by bylo nutné znovu zavést systém.

Soubor /etc/inittab je trochu složitější. Začneme tedy s jednoduchým příkladem konfigurace programu getty. Jednotlivé řádky souboru /etc/inittab sestávají ze čtyř polí oddělených dvoj-tečkou:

id:úrovně_běhu:akce:proces

Uvedené položky budou popsány níže. Soubor /etc/inittab může kromě řádkových záznamů obsahovat i prázdné řádky a řádky začínající znakem #. Ty proces init ignoruje. id

První položka identifikuje každý z řádků konfiguračního souboru. U řádků procesů getty se tak blíže specifikuje terminál, který daný proces obsluhuje (rozlišuje se číslem následujícím za příslušným názvem speciálního souboru /dev/tty). V řádcích pro ostatní procesy nemá toto pole žádný význam (kromě jeho omezení v délce), avšak mělo by být v celém souboru jedinečné.

úrovně_běhu

Úrovně běhu systému, které připadají pro daný řádek (proces) v úvahu. Úrovně se zadávají jako číslice bez oddělovače. Jednotlivé úrovně běhu systému budou popsány v následujícím odstavci.

akce

Akce, jež se má provést, například respawn (opakovaně spustí příkaz, jenž je uveden v dalším poli pokaždé, když je z různých důvodů ukončen) nebo once (spustí daný příkaz jenom jednou).

proces

Příkaz, který se má spustit.

Chcete-li spustit program getty pro první virtuální terminál (/dev/tty1) ve všech běžných víceu-živatelských úrovních běhu (2–5), vložte do souboru /etc/inittab tento řádek:

1:2345:respawn:/sbin/getty 9600 tty1

První pole identifikuje řádek pro zařízení /dev/tty1. Druhá položka určuje, že proces uvedený v posledním poli lze spouštět na úrovni běhu systému číslo 2, 3, 4 a 5. Třetí pole znamená, že tento příkaz by měl být vždy opakovaně spuštěn poté, co se ukončí (aby se po odhlášení uživa-tele mohl přihlásit kdokoliv jiný). V posledním poli je příkaz, který spouští proces getty pro první virtuální terminál.

Různé verze programu getty se spouštějí různě. Podívejte se na příslušnou manuálovou stránku a ověřte si, že čtete správnou stránku. Velmi často se v distribucích používá například mingetty.
Když budete potřebovat přidat do systému další terminály nebo modemové linky pro příchozí volání, měli byste rozšířit soubor /etc/inittab o další řádky, vždy jeden pro každý terminál, respektive modemovou linku. Více informací najdete v manuálových stránkách init, inittab a getty.

Nespustí-li se úspěšně příkaz uvedený v souboru /etc/inittab a je-li v konfiguraci procesu init nastavený jeho restart (akce respawn), bude proces zabírat značnou část systémových zdrojů. Pro-ces init totiž požadovaný proces spustí, ten se neprovede úspěšně a ukončí se, proces init jej opa-kovaně spustí, proces se ukončí, proces init jej spustí, proces se ukončí a tak dál, až donekoneč-na. Této situaci se předchází tím, že si proces init vede záznamy o tom, jak často se pokoušel určitý příkaz spustit. Je-li frekvence opakovaných pokusů o spuštění procesu příliš vysoká, pro-ces init před dalším pokusem o provedení příkazu vyčká pět minut.

Úrovně běhu systému

Úrovní běhu systému se rozumí určitý stav procesu init i celého systému. Tento stav určuje, které ze služeb se poskytují. Jednotlivé úrovně se rozlišují čísly. Někteří správci systémů pomocí těchto úrovní určují, které subsystémy se spustí, zdali například poběží X, jestli bude systém připojený k síti a podobně. Jiní dávají přednost inicializaci všech subsystémů na všech uživatelsky defino-vaných úrovních, popřípadě některé ze subsystémů spouští a zastavují samostatně bez toho, že by se měnily úrovně běhu systému. To proto, že počet úrovní je poměrně malý a řízení konfigurace takovýchto systémů pomocí jednotlivých uživatelsky definovaných úrovní běhu systému je tedy příliš hrubé. Správce systému se v této otázce musí rozhodnout sám, avšak nejjednodušší bude řídit se způsobem, jenž implementuje distribuce, ze které byl systém Linux instalován.

V následující tabulce jsou uvedeny různé úrovně běhu, jak jsou definovány ve většině linuxových distribucí. Nicméně, úrovně 2 až 5 si může každý uživatel změnit, jak uzná za vhodné.

0 Zastavení systému.

1	Jednouživatelský režim (pro administraci systému).
2	Lokální víceuživatelský se sítí, avšak bez síových služeb (např. NFS).
3	Plný víceuživatelský se sítí.
4	Nepoužito.
5	Plný víceuživatelský se sítí a s X Window (GUI).
6	Znovuzavedení systému.

Čísla úrovní běhu

Služby, které se spouštějí v čase, jsou určeny obsahem adresářů rcN.d. Většina distribucí má tyto adresáře umístěné buď v /etc/init.d/rcN.d nebo v /etc/rcN.d. (N nahraďte číslem úrovně běhu.)

V každé úrovni běhu naleznete řadu odkazů, které ukazují na startovací skripty umístěné v/etc/init.d. Jména těchto odkazů začínají buď písmeny K nebo na S, za nimiž následuje číslo. Začíná-li jméno odkazu na S, znamená to, že služba bude spuštěna, jakmile se dostanete na pří-slušnou úroveň. Pokud začíná na K, služba bude zrušena (pokud běží).

Číslo následující za KaSurčuje pořadí, v jakém budou skripty spouštěny. Uvedeme si příklad, jak může takový soubor /etc/init.d/rc3.d vypadat.

ls -l /etc/init.d/rc3.d lrwxrwxrwx 1 root root 10 2004-11-29 22:09 K12nfsboot -> ../nfsboot lrwxrwxrwx 1 root root 6 2005-03-29 13:42 K15xdm -> ../xdm

lrwxrwxrwx 1 root root 9 2004-11-29 22:06 S01random -> ../random lrwxrwxrwx 1 root root 11 2005-03-01 11:56 S02firewall -> ../firewall lrwxrwxrwx 1 root root 10 2004-11-29 22:34 S05network -> ../network lrwxrwxrwx 1 root root 9 2004-11-29 22:07 S06syslog -> ../syslog lrwxrwxrwx 1 root root 10 2004-11-29 22:09 S08portmap -> ../portmap

lrwxrwxrwx 1 root root 6 2004-11-29 22:09 S10nfs -> ../nfs lrwxrwxrwx 1 root root 12 2004-11-29 22:40 S12alsasound -> ../alsasound

lrwxrwxrwx 1 root root 7 2004-11-29 22:10 S12sshd -> ../sshd

lrwxrwxrwx 1 root root 7 2004-12-02 20:34 S13cups -> ../cups

powersaved

lrwxrwxrwx 1 root root 10 2004-11-29 22:10 S14postfix -> ../postfix lrwxrwxrwx 1 root root 6 2005-02-04 13:27 S14smb -> ../smb lrwxrwxrwx 1 root root 7 2004-11-29 22:10 S15cron -> ../cron lrwxrwxrwx 1 root root 8 2004-12-22 20:35 S15smbfs -> ../smbfs <zkráceno>

Úrovně běhu systému se konfigurují v souboru /etc/inittab řádkem podobným tomuto:

l2:2:wait:/etc/init.d/rc 2

V prvním poli je uvedeno libovolné návěští, druhé pole znamená, že tento záznam platí pro úro-veň běhu systému číslo 2. Třetí položka (pole wait) říká procesu init, aby spustil příkaz uvedený ve čtvrtém poli jenom jednou, a to při startu dané úrovně běhu systému, a pak vyčkal, než se pří-kaz provede. Samotný příkaz /etc/init.d/rc pak spustí všechny procesy a příkazy, které jsou potřebné pro spuštění a ukončení služeb, jimiž se implementuje úroveň běhu číslo 2.

Všechnu dřinu spojenou s nastavováním určité úrovně běhu dělá samotný příkaz uvedený ve čtvr-tém poli záznamu. Spouští služby, které zatím neběží, a pozastaví ty, které by na nové úrovni běhu již neměly být poskytovány. Záleží na konkrétní distribuci operačního systému Linux, který z pří-kazů bude v posledním políčku souboru /etc/inittab přesně uveden a které úrovně běhu budou v této konfiguraci implementovány.

Když proces init startuje, hledá ten řádek v souboru /etc/inittab, jenž specifikuje implicitní úroveň běhu systému:

id:2:initdefault:

Proces init lze také požádat o to, aby se spustil v jiné než běžné úrovni běhu, a to tak, že pře-dáte jádru systému řádkový parametr single nebo emergency. Parametry lze jádru předat napří-klad programem LILO. Tímto způsobem spustíte systém na úrovni 1.

Když už systém běží, lze změnit aktuální úroveň běhu příkazem telinit. V případě, že se úroveň běhu systému mění, proces init spouští ten příkaz v souboru /etc/inittab, jenž odpovídá nové úrovni běhu systému.

Zvláštní konfigurace v souboru /etc/inittab

Soubor /etc/inittab má některé speciální funkce, jež umožňují procesu init reagovat i na někte-ré zvláštní situace. Tyto speciální

funkce definují zvláštní klíčová slova ve třetím poli záznamu kon-figuračního souboru. Několik příkladů:

powerwait

Umožní procesu init v případě výpadku napájení zastavit systém. Předpokládá se, že systém používá záložní zdroj UPS a software, jenž sleduje UPS a informuje proces init o případném výpadku napájení.

ctrlaltdel

Nařizuje procesu init znovu zavést systém, když uživatel současně zmáčkne klávesy Control+Alt+Delete. Uvědomte si, že správce systému může reakci na tuto kombinaci nastavit tak, že se místo rebootu provede nějaká jiná akce, že se například – zvlášť když má k systému pří stup širší veřejnost – tato klávesová zkratka ignoruje. (Nebo se spustí program nethack.) sysinit

Příkaz spouštěný při startu systému. Typicky se takto zajistí například smazání adresáře /tmp.

Výše uvedený seznam není úplný. Další možnosti i podrobnosti týkající se těch, o kterých se zmi-ňujeme, uvádí manuálová stránka souboru inittab.

Zavádění systému v jednouživatelském režimu

Důležitou úrovní běhu systému je takzvaný jednouživatelský režim (úroveň běhu číslo 1), v němž může počítač používat pouze správce systému. V tomto režimu běží jenom minimum systémových služeb (včetně možnosti přihlášení do systému). Jednouživatelský režim je nutný pro některé úlohy spojené s údržbou systému. Například kontrola konzistence svazku /usr programem fsck vyžaduje, aby byla disková oblast se souborovým systémem odpojená. Toho ale nelze dosáhnout, pokud nejsou ukončeny téměř všechny systémové služby.

Běžící systém lze přepnout do jednouživatelského režimu příkazem telinit a požadavkem na pře-chod do úrovně běhu 1. Při zavádění systému lze do jednouživatelského režimu přejít zadáním parametru single nebo emergency na příkazové řádce jádra systému. Jádro předá parametry pří-kazové řádky procesu init. Proces init podle tohoto parametru pozná, že nemá použít implicitní úroveň běhu. (Způsob, kterým se zadává parametr příkazové řádky jádra systému, je podmíněn způsobem, jakým se zavádí operační systém.)

Někdy je potřeba zavést systém v jednouživatelském režimu například proto, aby bylo možné ručně spustit program fsck dřív, než se systém pokusí připojit poškozený systém souborů /usr nebo se s ním pokusí jiným způsobem manipulovat. Jakékoliv aktivity na defektním svazku jej s největší pravděpodobností poškodí ještě více, proto by se měla kontrola programem fsck udě-lat co nejdříve.

Zaváděcí skripty, které spouští proces init, automaticky přechází do jednouživatelského režimu pokaždé, když automatická kontrola programu fsck při zavádění systému neproběhne úspěšně. Pokouší se tak zabránit systému použít souborový systém, jenž je poškozený natolik, že jej nelze automaticky opravit zmiňovaným programem fsck. Takovéto poškození svazku je relativně málo frekventované a jeho příčinou bude pravděpodobně mechanické poškození pevného disku, pří-padně nějaká chyba v experimentální verzi jádra systému. Bude ale lepší, když budete jako správ-ci systému připraveni i na tuto situaci.
Správně nakonfigurovaný systém se před spuštěním příkazového interpretu v jednouživatelském režimu zeptá na přístupové heslo superuživatele. Je to důležité bezpečnostní opatření, protože jinak by bylo možné jednoduše zadat vhodný parametr příkazové řádky zavaděči systému LILO a dostat se tak k systému s oprávněním superuživatele. (Takto nastavený systém se samozřejmě nezavede, když bude důsledkem defektů na systémovém svazku soubor /etc/passwd poškoze-ný. Pro tento případ je dobré mít někde po ruce zaváděcí diskety.)

Přihlašování a odhlašování

Nestojím o to být členem klubu, který přijímá lidi, jako jsem já. (Groucho Marx)

Tato kapitola popisuje, co se v systému děje poté, když se uživatel přihlásí nebo odhlásí. Dále budou detailněji popsány různé interakce některých procesů běžících na pozadí, logovací soubo-ry používané při zahajování a ukončování sezení, konfigurační soubory a další.

Přihlašování přes terminály

V kapitole „Přihlášení z terminálů“ je popsáno, co se děje při přihlášení uživatele do systému přes terminál. V prvním kroku si

proces init ověří, zda běží program getty pro dané terminálové spo-jení (nebo konzolu). Program getty sleduje terminál a čeká na uživatele, jenž by mu sdělil, že se chce přihlásit do systému (obvykle tím, že stiskne některou klávesu na klávesnici terminálu). Když proces getty zjistí, že uživatel něco napsal na klávesnici, vypíše na obrazovku uvítací zprávu. Ta je uložená v souboru /etc/issue. Pak vyzve uživatele, aby zadal své uživatelské jméno, a nako-nec spustí program login. Program login dostane zadané uživatelské jméno jako parametr a následně vyzve uživatele, aby zadal přístupové heslo. Je-li heslo zadáno správně, program login spustí příkazový interpret vybraný podle nastavení konfigurace pro přihlášeného uživatele. V opačném případě se program login jednoduše ukončí a tím se ukončí i celý proces přihlašo-vání (většinou až poté, co uživatel dostane další možnost zadat správné uživatelské jméno a pří-stupové heslo). Proces init rozpozná, že byla procedura přihlašování ukončena, a spustí pro daný terminál novou instanci programu getty.

Je důležité si uvědomit, že jediným novým procesem je ten, jenž vytvoří proces init (použitím systémového volání fork). Procesy getty a login nahrazují právě tento nový proces (systémovým voláním exec).

V případě přihlašování po sériových linkách se pro sledování aktivity uživatelů používá zvláštní program proto, že někdy může být (a tradičně bývá) poměrně složité zjistit, kdy je terminál po nečinnosti opět aktivní. Program getty se rovněž přizpůsobuje přenosové rychlosti a dalším nasta-vením konkrétního spojení. Takovéto změny parametrů připojení jsou obzvlášť důležité v přípa-dě, že systém odpovídá na příchozí modemové žádosti o připojení, kdy se přenosové parametry běžně mění případ od případu.

V současnosti se používá několik různých verzí programů getty a proces init. Mají samozřejmě své výhody i nevýhody. Je dobré si přečíst dokumentaci k verzím, které jsou součástí vašeho systé-mu, ale rozhodně neuškodí ani informace o jiných verzích. Další dostupné verze programu lze vyhledat pomocí „Mapy programového vybavení pro Linux“ (The Linux Software Map). V případě, že nemusíte obsluhovat příchozí volání se žádostmi o přihlášení, nebudete se pravděpodob-ně muset programem getty zabývat, avšak podrobnější informace o programu proces init pro vás budou i nadále důležité.

Přihlášení prostřednictvím sítě

Dva počítače, které jsou zapojeny v jedné síti, jsou obvykle propojeny jediným fyzickým kabelem. Když spolu stanice prostřednictvím sítě komunikují, programy, které běží na každé z nich a podí-lejí se na vzájemné komunikaci, jsou propojeny *virtuálními spojeními*, tedy jakousi sadou imagi-nárních kabelů. Když spolu aplikace na obou koncích virtuálního spojení komunikují, mají pro sebe vyhrazenou vlastní „linku“. Protože tato linka není skutečná, pouze imaginární, mohou ope-rační systémy na obou počítačích vytvořit i několik virtuálních spojení sdílejících tutéž fyzickou linku. Takto spolu může s využitím jediného kabelu komunikovat několik programů bez toho, že by o ostatních spojeních věděly nebo se o ně nějakým jiným způsobem staraly. Stejné fyzické médium může být sdíleno i několika počítači. Když pak existuje virtuální spojení mezi dvěma sta-nicemi, další počítače, které se komunikace neúčastní a sdílí tutéž fyzickou linku, toto spojení ignorují.

Toto byl komplikovaný a možná až příliš odtažitý popis reality. Měl by ale stačit k pochopení důle-žitého rozdílu mezi přihlášením prostřednictvím sítě a normálním přihlášením přes terminál. Vir-tuální spojení se vytvoří v případě, že existují dva programy na různých stanicích a přejí si spolu komunikovat. Vzhledem k tomu, že je principiálně možné připojit se z kteréhokoliv počítače v síti na kterýkoliv jiný, existuje velké množství potenciálních virtuálních spojení. Díky tomu není prak-tické spouštět proces getty pro každé potenciální síťové přihlášení do systému.

Proto také existuje jediný proces inetd (odpovídající procesu getty), který obsluhuje *všechna* síťo-vá připojení. V případě, že proces inetd zaregistruje žádost o připojení ze sítě (tedy zaregistruje navázání nového virtuálního spojení s některým jiným počítačem zapojeným v síti), spustí nový proces obsluhující toto jediné přihlášení. Původní proces je nadále aktivní a dále čeká na nové požadavky o připojení. V současnosti se používá spíše novější xinetd.

Aby to nebylo až tak jednoduché, existuje pro připojení ze sítě víc komunikačních protokolů. Dva nejstarší jsou telnet a rlogin. Kromě připojení do systému existuje i mnoho dalších druhů virtu-álních spojení, která lze mezi počítači v síti navázat (síťové služby FTP, Gopher, HTTP a další). Bylo by neefektivní mít zvláštní proces, jenž by sledoval žádosti o navázání spojení pro každý typ připojení (službu). Místo toho existuje jediný proces, který umí rozeznat typ spojení a spustit správný program, jenž pak poskytuje odpovídající služby. Tímto procesem je právě proces inetd. Podrobnější informace uvádí *Příručka správce sítě*.

Co dělá program login

Program login se stará o autentizaci uživatele (kontroluje, zda bylo zadáno správné uživatelské jméno a přístupové heslo) a počáteční nastavení uživatelského prostředí nastavením oprávnění pro sériovou linku a spuštěním interpretu příkazů.

Částí procedury úvodního nastavení uživatelského prostředí je i vypsání obsahu souboru /etc/motd (zkratka pro „message of the day“ – zpráva pro tento den) a kontrola nově příchozí elektronické pošty. Tyto kroky lze zakázat vytvořením souboru nazvaného .hushlogin v domov-ském adresáři uživatele.

Existuje-li soubor /etc/nologin, jsou přihlášení do systému zakázána. Tento soubor je typicky vytvářen příkazem shutdown nebo příbuznými programy. Program login kontroluje, jestli tento soubor existuje, a v případě, že je tomu tak, odmítne akceptovat přihlášení a předtím, než se defi-nitivně ukončí, vypíše obsah tohoto souboru na terminál.

Program login rovněž zapisuje všechny neúspěšné pokusy o přihlášení do systémového logu (pomocí programu syslog). Rovněž

zaznamenává úspěšné i neúspěšné pokusy o přihlášení super-uživatele. Oba druhy záznamů jsou užitečné při pátrání po případných „vetřelcích“.

Momentálně přihlášení uživatelé jsou zapsáni v seznamu /var/run/utmp. Tento soubor je platný jenom do dalšího znovuzavedení nebo zastavení systému, protože v průběhu zavádění systému se jeho obsah vymaže. Jinak jsou v souboru /var/run/utmp kromě seznamu všech přihlášených uživatelů a používaných terminálů (nebo síťových spojení) uvedeny i další užitečné informace. Příkazy who, w a další podobné se dívají právě do souboru /var/run/utmp a zjišťují, kdo je k systé-mu připojený.

Všechna úspěšná přihlášení jsou zaznamenána do souboru /var/log/wtmp. Tento soubor se může bez omezení zvětšovat, proto je potřeba jej pravidelně mazat (například po týdnu) zadáním úkolu démonu cron. Soubor wtmp lze procházet příkazem last.

Oba soubory utmp i wtmp mají binární formát (viz manuálová stránka utmp), takže je nelze pro-hlížet bez speciálních programů.

X a xdm

X implementují přihlášení prostřednictvím procesu – správce displeje – xdm. Nové distribuce dávají spíše přednost správcům kdm nebo gdm z prostředí KDE/Gnome.

Řízení přístupu

Databáze uživatelů je tradičně uložena v souboru /etc/passwd. Některé systémy používají tak-zvaná *stínová hesla*. Přesouvají uživatelská přístupová hesla ze souboru /etc/passwd do soubo-ru /etc/shadow. Sítě s velkým počtem počítačů, ve kterých se informace o uživatelských účtech sdílí pomocí systému NIS nebo nějakou jinou metodou, mohou databázi uživatelů automaticky kopírovat z jediného centrálního počítače na všechny ostatní stanice.

Databáze uživatelů obsahuje nejenom hesla, ale i některé další informace o uživatelích, například jejich skutečná jména, domovské adresáře a interprety příkazů, jež se implicitně spouští po přihlášení.

Je potřeba, aby byly tyto informace o uživatelích v systému obecně dostupné a aby si je mohl každý přečíst. Kvůli tomu se heslo ukládá v zakódovaném tvaru. Má to ale jeden háček. Každý, kdo má přístup k databázi uživatelů, může s pomocí různých kryptografických metod zkusit hesla uhodnout i bez toho, že by se musel přihlásit k hostitelskému počítači. Systém stínových hesel se snaží zamezit možnosti prolomení přístupových hesel tím, že se přesouvají do jiného souboru, jenž je přístupný pouze superuživateli (hesla se i tak ukládají v zakódovaném tvaru). Avšak s pozdější instalací systému stínových hesel na systému, který jej nepodporuje, mohou vznikat různé potíže.

Ať tak nebo onak, z bezpečnostních důvodů je důležité pravidelně ověřovat, jestli jsou všechna přístupová hesla používaná v systému netriviální, tedy taková, aby nebylo lehké je uhodnout. Lze použít například program crack, jenž zkouší hesla v /etc/passwd dekódovat. Heslo, které se mu podaří uhodnout, nelze podle výše uvedeného považovat za spolehlivé. Program crack mohou samozřejmě zneužít i případní vetřelci, ale správci systému může jeho pravidelné používání pomo-ci preventivně omezit výběr nevhodných přístupových hesel. K volbě netriviálního přístupového hesla lze uživatele donutit i programem passwd. To je metoda, která je efektivnější především z hlediska zatížení procesoru, protože zpětná analýza zašifrovaných hesel programem crack je výpočetně o hodně náročnější.
Databáze skupin uživatelů je uložena v souboru /etc/group, u systémů se stínovými hesly pří-
padně v souboru /etc/gshadow. Uživatel root se obvykle nemůže přihlásit z kteréhokoliv terminálu nebo počítače v síti, pouze z terminálu uvedeného v seznamu /etc/securetty. Pak je nutné mít k některému z těchto ter-minálů fyzický přístup. Avšak takovéto bezpečnostní opatření nelze považovat za dostatečné, pro-tože je možné přihlásit se z kteréhokoliv jiného terminálu jako běžný uživatel a pro změnu uži-vatelských oprávnění použít příkaz su.

Spouštění interpretu příkazů

Při startu každý příkazový interpret automaticky spouští jeden či více předem určených souborů. Různé interprety spouští různé konfigurační soubory. Podrobnější informace najdete v dokumen-taci k jednotlivým typům interpretů.

Většina příkazových interpretů nejdříve spustí některý globální soubor, například interpret Bour-ne shell (/bin/sh) a jeho klony spouští soubor /etc/profile, až poté spustí soubor .profile, jenž je uložen v uživatelově domovském adresáři. Soubor /etc/profile umožňuje správci syste-mu nastavit běžné, implicitní uživatelské prostředí, například nastavením proměnné PATH, tak aby zahrnovalo kromě obvyklých i lokální adresáře s příkazy. Soubor .profile zase umožňuje kaž-dému z uživatelů upravit si předem nastavené běžné prostředí podle svého vlastního vkusu.

Správa uživatelských

účtů

Jaký je rozdíl mezi správcem systému a překupníkem drog? Žádný, oba měří své zboží v kilech a oba mají své klienty.
(Starý a otřelý počítačový vtip)

Tato kapitola popisuje způsob vytváření nových uživatelských účtů, změny vlastností těchto účtů a způsoby jejich odstraňování. Různé distribuce systému Linux používají pro tyto úkoly různé nástroje.

Co je to účet?

Používá-li počítač více lidí, je obvykle nutné mezi jednotlivými uživateli rozlišovat. Například proto, aby jejich osobní soubory a data byly osobními v pravém smyslu slova. Identifikace uživa-telů je ale důležitá i v případě, že systém využívá v jednom okamžiku pouze jedna osoba, což se týká převážné většiny mikropočítačů. Proto má každý uživatel systému přiděleno jednoznačné uži-vatelské jméno, které zadává při každém přihlášení.

Avšak pojem „účet“ je poněkud širší a zahrnuje – kromě uživatelského jména – i některé další informace o uživateli. Pojmem *uživatelský účet* se označují všechny soubory, zdroje a informace, jež se vztahují k danému uživateli. Běžně se termín „účet“ spojuje s bankovním sektorem. V komerčním systému se kolem každého účtu skutečně točí nějaké peníze. Ty mohou z účtu mizet různou rychlostí, podle toho, jak moc uživatel systém zatěžuje. Tak například diskový prostor lze ohodnotit cenou za megabajt uložených dat a den, čas procesoru může mít určitou cenu za sekun-du využití a podobně.

Vytváření uživatelských účtů

Samotné jádro systému Linux považuje uživatele systému za pouhé číslo. Každého uživatele lze totiž identifikovat podle jednoznačného celého čísla, takzvaného *identifikačního čísla uživatele* (anglicky *user ID*, zkráceně *UID*). Je tomu tak proto, že počítač umí zpracovat čísla rychleji a jed-nodušeji než jména v textové formě. Jména v textové podobě (*uživatelská jména*) se udržují ve zvláštní databázi mimo vlastní jádro. Tato databáze obsahuje též další informace o uživatelích systému.
Když potřebujete vytvořit nový uživatelský účet, musíte přidat informace o novém uživateli do této uživatelské databáze a vytvořit pro něj vlastní domovský adresář. Kromě toho by měl každý nový uživatel absolvovat školení. Je též vhodné nastavit pro nové uživatele přiměřené počáteční nastavení prostředí.

Většina distribucí systému Linux se dodává s programy pro vytváření uživatelských účtů. Správce má dokonce k dispozici hned několik takovýchto programů. Dvě varianty, jejichž uživatelským rozhraním je příkazová řádka, jsou programy adduser a useradd. Existují i nástroje s grafickým uživatelským rozhraním. Ať už se jako správce systému rozhodnete pro kterýkoliv z programů, oceníte jejich hlavní přínos, tedy to, že omezují manuální práci s nastavováním na minimum, či dokonce úplně. I když na vás při administraci uživatelských účtů čeká mnoho dost spletitých detailů, díky těmto nástrojům vypadá jednoduše. Postup při „ručním“ zakládání nových uživatel-ských účtů uvádí kapitola „Manuální vytváření účtů“.

Soubor /etc/passwd a další soubory

Základní databází uživatelů v systému Unix je textový soubor /etc/passwd (označovaný jako *password file*), v němž jsou uvedena platná uživatelská jména a další k nim přidružené informa-ce. Každému uživateli odpovídá v souboru jeden záznam – řádek, který je rozdělen na sedm polí, jejichž oddělovačem je dvojtečka. Význam jednotlivých položek je následující:

Uživatelské jméno.
Heslo v zakódované podobě.
Identifikační číslo uživatele.
Identifikační číslo skupiny.
Skutečné jméno uživatele, případně popis účtu.
Domovský adresář.
Příkazový interpret (nebo program), který se spustí po přihlášení.

Formát jednotlivých políček je podrobněji popsán v manuálové stránce souboru passwd. Každý uživatel systému má k souboru /etc/passwd přístup (může jej číst). Může tedy například zjistit přihlašovací jména ostatních uživatelů. To ale znamená, že jsou všem přístupná i hesla ostat-ních uživatelů (druhé pole každého záznamu). Hesla uložená v souboru /etc/passwd jsou zakó-dovaná, takže teoreticky nevzniká žádný problém. Avšak použitý kódovací algoritmus lze prolo-mit, zvlášť je-li zvolené heslo jednoduché

(například krátké slovo, jež lze najít v nějakém slovní-
ku, jméno nebo příjmení uživatele a podobně). Proto z hlediska bezpečnosti není dobré mít hesla uložená přímo v souboru /etc/passwd. Řada systémů Linux používá systém takzvaných *stínových hesel*. Jde o alternativní způsob uložení uživatelských přístupových hesel, jež se zašifrovaná ukládají do jiného souboru (/etc/shadow), který může číst pouze superuživatel. Soubor /etc/passwd pak obsahuje v druhém poli pouze speciální znak. Programy, které si potřebují ověřit totožnost uživatele, mají propůjčená přístupová práva vlastníka souboru. Běžné programy, které používají pouze některé z dalších položek sou-boru /etc/passwd, přístup k heslům uživatelů systému nemají.

Výběr čísel uživatelského ID a ID skupiny

Ve většině systémů nezáleží na tom, jaké jsou hodnoty UID a GID. Když ale používáte síťový sou-borový systém NFS, musíte mít stejná UID a GID na všech systémech v síti. To proto, že i systém NFS identifikuje uživatele podle hodnoty UID. Jestliže systém NFS nepoužíváte, můžete vybírat
identifikační čísla uživatele a skupiny podle automatického návrhu některého z nástrojů pro sprá-vu uživatelských účtů. Když v síti využíváte NFS, budete si muset zvolit některý z mechanismů synchronizace informací
o uživatelských účtech. Jednou z možností je systém NIS – Network Information Service.

Měli byste se však vyvarovat opakovaného používání číselných UID (a textových jmen), neboť nový uživatel by mohl získat přístup do souborů původního uživatele (nebo pošty či čehokoli jiného).

Nastavení uživatelského prostředí: adresář /etc/skel

Když jste již pro nového uživatele vytvořili vlastní domovský adresář, můžete nastavit vlastnosti uživatelského prostředí tak, že do domovského adresáře nového uživatele nakopírujete některé soubory z adresáře /etc/skel. Správce systému totiž může v adresáři /etc/skel vytvořit konfi-gurační soubory, jež novým uživatelům vytvoří příjemné základní uživatelské prostředí. Admini-strátor může například vytvořit soubor /etc/skel/.profile, v němž lze nastavením proměnné prostředí EDITOR vybrat některý z textových editorů, jenž by měl pro méně zkušené uživatele přátelské ovládání.

Avšak obvykle se doporučuje mít v adresáři /etc/skel co nejméně souborů, protože by jinak bylo téměř nemožné upravit na větších víceuživatelských systémech při každé změně konfigurač-ní soubory v již existujících uživatelských adresářích. Když se například změní název onoho uži-vatelsky příjemného editoru, všichni současní uživatelé si musí upravit svůj vlastní soubor .pro-file. Správce systému by se to mohl pokusit udělat automaticky, například pomocí skriptu, ale takovéto akce téměř pravidelně končí tak, že se nechtěně přepíše či poškodí nějaký jiný soubor.

Vždy když to situace dovolí, je lepší nastavovat globální konfigurace v globálních souborech, jako je /etc/profile. Pak lze nastavení pohodlně upravovat bez toho, že by se muselo měnit vlastní nastavení jednotlivých uživatelů systému.

Manuální vytváření účtů

Nový uživatelský účet lze vytvořit ručně tímto postupem:

Upravte soubor /etc/passwd příkazem vipw tak, že do souboru hesel přidáte nový řádek pro nový uživatelský účet. Je nutné dodržovat správnou syntaxi. *Není vhodné upravovat tento soubor přímo běžným editorem!* Program vipw soubor /etc/passwd uzamkne, takže jej ostatní programy nemohou změnit. Do pole pro heslo vložte znak *, takže zatím nebu-de možné se prostřednictvím tohoto účtu do systému přihlásit.
Je-li třeba vytvořit i novou pracovní skupinu, upravte soubor /etc/group rovněž progra-mem vigr.
Příkazem mkdir pro nového uživatele vytvoříte domovský adresář.
Do nově vytvořeného domovského adresáře nakopírujte konfigurační soubory z adresáře /etc/skel.
Příkazy chown a chmod upravte jejich vlastníka a přístupová práva. Užitečný je v tomto případě jejich parametr -R. Správná přístupová práva se mohou trochu lišit, a to podle typického využití toho kterého systému, nicméně příkazy v níže uvedeném příkladu obvykle vyhovují většině případů:

```
cd /home/newusername
chown -R username:group .
chmod -R go=u,go-w .
chmod go= .
```

- Zadejte heslo programem passwd.

Poté co bylo v posledním kroku nastaveno heslo, bude nový účet přístupný. Heslo by se skuteč-ně mělo nastavovat až v posledním kroku, jinak by se mohl uživatel nepozorovaně přihlásit do systému například v době, kdy kopírujete konfigurační soubory.

Někdy je potřeba vytvořit „falešný“ účet, který se nebude používat pro přihlašování běžných uži-vatelů. Například při

konfigurování anonymního serveru FTP je výhodné vytvořit účet s uživatel-ským jménem ftp. Pak si ze serveru může stahovat soubory kdokoliv a pro budoucí (anonymní) klienty se nemusí zřizovat vlastní uživatelské účty. V takovýchto případech obvykle není třeba nastavovat heslo (vynechá se poslední krok výše uvedeného postupu). Je lepší zřizovat takovéto anonymní účty bez nastaveného hesla. Pak k nim má totiž přístup pouze superuživatel.

Změny vlastností uživatelských účtů

Je několik příkazů, kterými lze měnit různé vlastnosti uživatelských účtů (tedy příslušných polo-žek v souboru /etc/passwd):

chfn

Mění pole, ve kterém je uloženo skutečné jméno uživatele.

chsh

Mění nastavený příkazový interpret.

passwd

Mění přístupové heslo.

Superuživatel může pomocí těchto programů změnit vlastnosti kteréhokoliv účtu. Ostatní neprivi-legovaní uživatelé mohou měnit pouze vlastnosti svého vlastního účtu. V některých případech je vhodnější běžným uživatelům zakázat možnost používat uvedené příkazy (programem chmod), například v případě, že systém používá větší počet méně zkušených začátečníků.

Ostatní změny položek souboru /etc/passwd se musí dělat ručně. Když například potřebujete změnit uživatelské jméno, musíte přímo upravit databázi uživatelů /etc/passwd (připomínáme, že pouze příkazem vipw). Analogicky, když potřebujete přidat či odebrat uživatele z nebo do některé z pracovních skupin, musíte upravit soubor /etc/group (příkazem vigr). Avšak takové-to úkoly se dělají zřídka a musí se dělat opatrně, protože když například změníte některému z uži-vatelů jeho uživatelské jméno, nebude mu docházet elektronická pošta a musíte pro něj vytvořit poštovní alias.

Zrušení uživatelského účtu

Potřebujete-li zrušit uživatelský účet, smažte nejdříve všechny soubory, které patří uživateli ruše-ného účtu, včetně poštovní schránky, aliasů pro elektronickou poštu, tiskových úloh, úkolů spouš-těných démony cron a at, a všechny další odkazy na tohoto uživatele. Pak odstraňte odpovídají-cí řádek v souborech /etc/passwd a /etc/group (nezapomeňte odstranit uživatelské jméno i ze všech skupin, do nichž byl uživatel zařazen). Předtím než začnete mazat vše ostatní, je lepší zaká-zat přístup k rušenému účtu (viz níže). Uživatel tak nebude mít možnost se připojit do systému v době, kdy je jeho účet odstraňován.

Nezapomeňte, že uživatelé systému mohou mít některé soubory uloženy i mimo svůj domovský adresář. Najdete je pomocí příkazu find:

find / -user *username*

Pamatujte na to, že když má váš systém velké disky, poběží výše uvedený příkaz dost dlouho. V případě, že máte v souborovém systému připojeny síťové disky, musíte dávat pozor, abyste tím nezpůsobili problémy s odezvou v síti nebo na serveru.

Součástí některých distribucí systému Linux jsou i zvláštní příkazy, které lze použít při rušení uži-vatelských účtů. Zkuste na vašem systému vyhledat programy deluser či userdel. Každopádně není zase tak složité to udělat ručně, navíc tyto programy nemusí najít a odstranit všechny sou-vislosti.

Dočasné zablokování uživatelského účtu

Občas je potřeba dočasně zablokovat přístup k některému z účtů bez toho, že by bylo nutné jej smazat například když uživatel nezaplatil poplatky za využívání systému nebo když má správce systému podezření, že neznámý útočník prolomil přístupové heslo uživatele některého účtu.

Nejvhodnějším způsobem zamezení přístupu k podezřelému účtu je zaměnit nastavený příkazovýinterpret na zvláštní program, který na terminál pouze vypíše nějaké hlášení. Když se pak kdo-koliv pokusí přihlásit do systému přes zablokovaný účet, neuspěje a dozví se proč. Zprávou lzesdělit uživateli, že se má spojit se správcem systému a domluvit se s ním na řešení vzniklého pro-blému.

Alternativou je změna uživatelského jména, případně hesla. Uživatel se tak ale nedozví, co se vlast

ně děje. A zmatení uživatelé znamenají i více práce pro správce systému. Nejjednodušší způsob, jak vytvořit program, který by blokoval přístup k účtu, je napsat skript proprogram tail:

#!/usr/bin/tail +2Tento účet byl z důvodů porušení bezpečnostních opatření zablokován.Volejte, prosím, číslo 555-1234 a vyčkejte příjezdu mužů

v černém.

Podle prvních dvou znaků (#!) pozná jádro systému, že zbytek řádku je příkaz, který je třeba spus-tit, aby se skript provedl. V tomto případě je to příkaz tail, jenž vypíše vše kromě prvního řádku na standardní výstup.

Je-li uživatel billg podezřelý, že porušuje bezpečnostní opatření, může správce systému udělat něco jako:

chsh -s /usr/local/lib/no-login/security billg # su – billg Tento účet byl z důvodů porušení bezpečnostních opatření zablokován. Volejte, prosím, číslo 555-1234 a vyčkejte příjezdu mužů v černém.

Pomocí příkazu su ve výše uvedeném příkladu se pochopitelně pouze testuje, zda je změna
původně nastaveného interpretu funkční.
Takovéto skripty by měly být uloženy ve zvláštním adresáři, aby jejich jména nekolidovala s pří
kazy jednotlivých uživatelů.

Zálohování

Hardware je nedeterministicky spolehlivý.
Software je deterministicky nespolehlivý.
Lidé jsou nedeterministicky nespolehliví.
Příroda je deterministicky spolehlivá.

Tato kapitola vysvětluje, proč, jak a kdy zálohovat a jak ze záloh obnovit data.

Jak je důležité míti zálohy

Vaše data mají určitou cenu. Jejich cena je daná hodnotou vašeho času a cenou úsilí potřebného pro nové vytvoření dat v případě jejich ztráty. To vše lze vyjádřit v penězích, nebo přinejmenším vyvážit zármutkem a slzami. Někdy již totiž ztracené informace nelze obnovit, například když data pocházejí z nějakých neopakovatelných experimentů. Vzhledem k tomu, že informace a data jsou v jistém slova smyslu investicí, měli byste tuto investici chránit a učinit kroky, jež by zabránily jejímu znehodnocení.

V zásadě existují čtyři důvody, jež by mohly vést ke ztrátě dat: závady hardwaru, chyby v pro-gramech, jednání lidí nebo přírodní katastrofy. I když je v současnosti hardware relativně spoleh-livý, ještě stále se může v podstatě bez příčiny porouchat. Pokud jde o ukládání dat, je nejkritič-tější součástí výpočetního systému pevný disk. Ve světě plném elektromagnetického šumu se blá-hově spoléhá na to, že miniaturní magnetická políčka na povrchu disku, ve kterých jsou cenné informace uloženy, zůstanou v neporušeném stavu. U programů o spolehlivosti prakticky nelze mluvit, robustní a spolehlivý software je spíše výjimkou než pravidlem. I lidé jsou dost nespoleh-liví. Buďto udělají nějakou chybu nebo jsou zlomyslní a zničí data úmyslně. Přírodu obecně bychom neměli považovat za zlo, ale i když je na nás hodná, může způsobit zkázu. Takže je malým zázrakem, když všechno funguje, jak má.

Řekli jsme, že zálohování je způsobem ochrany investic do dat a informací. Máte-li několik kopií dat, nestane se zase až tak moc, když se některá z verzí zničí (cenou za takovou ztrátu je pouze to, že data musíte obnovit ze zálohy).

Je důležité zálohovat správně. Jako všechno ostatní v reálném světě, dříve nebo později selžou i zálohy. Součástí správného zálohování je i to, že se kontroluje, jestli vůbec funguje. Jistě byste si nechtěli jednoho dne „všimnout“, že se zálohy nedělaly správně. Neštěstí v neštěstí – může se stát, že dojde k nějaké vážné havárii právě v okamžiku, kdy zálohujete a máte pouze jediné zálo-hovací médium. To se může rovněž poškodit a z úmorné práce zbude (někdy doslova) hromád-ka popela. Může se také stát, že si v momentě, kdy se pokoušíte obnovit data ze záloh, uvědo-míte, že jste zapomněli zálohovat něco důležitého, například databázi uživatelů systému, která může mít třeba 15 000 položek. To „nejlepší“ nakonec – všechny zálohy mohou fungovat bezchybně, ale v poslední známé páskové jednotce, jež ještě umí přečíst typ pásky, který používáte, je vědro vody.

Když totiž dojde na zálohy, je paranoia v popisu práce.

Výběr média pro zálohování

Co se týče zálohování, je nejdůležitějším rozhodnutím výběr médií. Musíte zvážit náklady, spo

lehlivost, rychlost, dostupnost a použitelnost.
Cena je poměrně důležitá, protože byste měli mít několikrát větší kapacitu zálohovacích médií, než je velikost zálohovaných dat. Levné médium je obvykle nezbytnost.

Extrémně důležitá je spolehlivost, protože poškozená záloha by rozbrečela i dospělého. Data ulo-žená na zálohovacích médiích musí vydržet bez poškození i několik let. I způsob, kterým médi-um používáte, má vliv na jeho spolehlivost. Například pevný disk je typicky velmi spolehlivémédium, ovšem z hlediska zálohování je to k ničemu, pokud jej máte ve stejném počítači, ze kte-rého zálohujete.

Rychlost většinou není příliš důležitá, pokud zálohování může probíhat automaticky. Když nemu-síte na zálohování dohlížet, pak nevadí, že trvá dejme tomu dvě hodiny. Ale naopak, nelze-li pro-vést zálohování v době, ve které je počítač jinak nevyužitý (například v průběhu noci), pak můžebýt i rychlost problémem.

Dostupnost je zjevně důležitá, protože nemůžete používat zálohovací médium, které není k dostá-ní. Méně zjevný je důležitý požadavek, aby bylo zálohovací médium dostupné i v budoucnu a pří-padně i pro jiné počítače než ty, které používáte dnes. Jinak se může stát, že nebudete schopnizálohy po nečekané události obnovit.

Použitelnost je nejvíce závislá na tom, jak často se zálohování provádí. Čím jednodušší je záloho

vání, tím lépe. Zálohování na zvolené médium nesmí být složité, pracné nebo otravné. Typickými alternativami jsou diskety a pásky. Diskety jsou velmi levné, celkem spolehlivé, ne pří-liš rychlé, velmi dostupné, ale ne příliš použitelné v případě velkého množství dat. Magneticképásky jsou levné i drahé, celkem spolehlivé, celkem rychlé, dost dostupné a – podle toho, jaká jejejich kapacita – z hlediska použitelnosti i docela pohodlné.

Existují i jiné možnosti. Obvykle nejsou nejlepší, pokud jde o jejich dostupnost, ale vznikne-li pro-blém, oceníte je více než ostatní možnosti. Řeč je o magneto-optických discích, jež kombinujídobré vlastnosti disket (jejich přímý, nesekvenční přístup k souborům, možnost rychlého obno-vování jednotlivých souborů) a magnetických páskových médií (zálohování velkého objemu dat).

Výběr nástroje pro zálohování

Existuje bezpočet nástrojů, jichž lze při zálohování využít. Mezi ty tradiční, používané při záloho-vání v systémech Unix, patří programy tar, cpio a dump. Kromě toho lze sáhnout i po velkém množství balíků programů třetích výrobců (šířených zdarma jako freeware i komerčně). Výběr médií pro zálohování má často vliv i na výběr nástroje.

Programy tar a cpio jsou podobné a z hlediska zálohování prakticky stejné. Oba programy umí ukládat soubory na pásky a obnovovat je, oba jsou schopné využívat téměř všechna média, pro-tože díky ovladačům jádra systému, které mají na starost obsluhu nízkoúrovňových zařízení, se takováto zařízení chovají vůči programům na uživatelské úrovni stejně. Některé unixové verze programů tar a cpio mohou mít problémy s neobvyklými soubory (symbolickými odkazy, speci-álními soubory, soubory s velmi dlouhou cestou a podobně), avšak verze pro operační systém

Linux by měly pracovat se všemi soubory korektně.Program dump se liší v tom, že přistupuje k souborovému systému přímo a ne prostřednictvímjeho služeb. Navíc byl napsán speciálně pro zálohování, kdežto programy tar a cpio jsou pri-márně určeny pro archivaci souborů, i když je lze s úspěchem použít i jako zálohovací nástroje.

Přímý přístup k souborovému systému má několik výhod. Umožňuje zálohovat soubory bez toho,že by to mělo vliv na jejich časové značky. U programů tar a cpio byste museli nejdřív připojitsouborový systém pouze pro čtení. Přímé čtení dat ze souborového systému je také efektivnějšív případech, kdy je potřeba zálohovat všechna data na disku, protože v tomto případě proběhnezálohování s výrazně menším pohybem čtecích hlav. Největší nevýhodou přímého přístupu je, žezálohovací program, který jej používá, je specifický pro každý typ souborového systému. Programdump pro Linux rozumí pouze souborovému systému ext2.
Program dump má navíc zabudovanou podporu úrovní zálohování, o kterých bude řeč později.U programů tar a cpio musí být implementovány pomocí jiných nástrojů jako gzip nebo bzip2.Srovnávání zálohovacích nástrojů třetích výrobců jde nad rámec této knihy. „Mapa programů proLinux“ (The Linux Software Map) uvádí výčet těch, které jsou jako freeware šířeny zdarma.

Jednoduché zálohování

Při proceduře jednoduchého zálohování se v prvním kroku vytvoří zálohy všech dat a v dalších krocích se pak zálohuje jenom to, co se od posledního zálohování změnilo. Prvnímu kroku se říká *úplné zálohování*, v dalších krocích se tvoří *inkrementální zálohy*. Úplné zálohování je obvykle pracnější než inkrementální, protože se na médium zapisuje víc dat, a proto se úplná záloha nemu-sí vždy vejít na jednu pásku (nebo disketu). Naopak, obnovování dat z inkrementálních záloh bývá pochopitelně mnohokrát pracnější než obnovování z úplných záloh. Nicméně inkrementální zálo-hování lze optimalizovat tak, že se vždy zálohují všechny změny od poslední úplné zálohy. Takto je sice o něco pracnější proces zálohování, ale pak by nemělo být nikdy potřeba obnovovat víc než jednu úplnou a jednu inkrementální zálohu.

Uveďme příklad, kdy chcete data zálohovat každý den a máte k dispozici šest páskových kazet, můžete první z nich použít (řekněme v pátek) pro uložení první úplné zálohy a pásky 2 až 5 pro inkrementální zálohování (od pondělí do čtvrtka). Pak vytvoříte novou úplnou zálohu na pásku číslo 6 (druhý pátek) a začnete znovu s inkrementálním zálohováním na pásky 2 až 5. Nepřepisuj-te pásku číslo jedna do doby, než se ukončí nové úplné zálohování – v jeho průběhu by se totiž mohlo stát něco neočekávaného. Poté co vytvoříte novou úplnou zálohu na pásce číslo 6, měli byste uschovat pásku 1 někam jinam pro případ, že by se ostatní zálohovací pásky zničily – např-klad při požáru. Takto vám v případě neočekávané události zůstane alespoň něco. Když pak potře-bujete vytvořit další úplnou zálohu, dojdete pro pásku číslo 1 a pásku číslo 6 necháte na jejím místě.

Máte-li víc než šest kazet, můžete ukládat na ty, jež jsou navíc, další úplné zálohy. Pokaždé když budete dělat úplnou zálohu, použijete tu nejstarší pásku. Takto budete mít úplné zálohy z něko-lika předchozích týdnů, což je dobré, když potřebujete obnovit například některý starší, omylem smazaný soubor, případně starší verzi nějakého souboru.

Zálohování programem tar

Pomocí programu tar lze jednoduše vytvořit úplnou zálohu tímto způsobem:

```
# tar –create –file /dev/ftape /usr/src tar: Removing leading / from absolute path names in the archive #
```

Ve výše uvedeném příkladu byla použita syntaxe programu tar verze GNU s dlouhými tvary pře-pínačů. Tradiční verze programu tar rozumí pouze jednopísmenným volbám. Verze GNU ale umí pracovat i se zálohami, které se nevejdou na jednu pásku nebo disketu, a s velmi dlouhými ces-tami k zálohovaným souborům. Ne všechny klasické verze programu tar tyto věci zvládají. (Ope-rační systém Linux používá výhradně program tar ve verzi GNU.)

Jestli se záloha nevejde na jednu pásku, je potřeba zadat přepínač --multi-volume (-M):

```
# tar -cMf /dev/fd0H1440 /usr/src tar: Removing leading / from absolute path names in the archive Prepare volume #2 for /dev/fd0H1440 and hit return: #
```

Nezapomeňte, že před zálohováním je potřeba diskety zformátovat. Stačí, když to uděláte v jiném okně, případně na jiném virtuálním terminálu ve chvíli, kdy program tar čeká na vložení další dis-kety.

Pokaždé když ukončíte zálohování, měli byste zkontrolovat, zda proběhlo správně. Zadejte příkaz tar s parametrem -compare (-d):

```
# tar –compare –verbose -f /dev/ftapeusr/src/usr/src/linuxusr/src/linux-1.2.10-includes/<zkráceno>#
```

Chyba v záloze znamená, že nic nepoznáte až do okamžiku, než budete potřebovat data ze zálo

hy obnovit.
Je-li kontrola záloh neúspěšná, nebudete moci v případě potřeby původní data z takovýchto záloh obnovit.

Inkrementální zálohování dělá program tar s parametrem -newer (-N):

```
# tar –create –newer ‘8 Sep 1995’ –file /dev/ftape /usr/src –verbosetar: Removing leading / from absolute path names in the archiveusr/src/usr/src/linux-1.2.10-includes/usr/src/linux-1.2.10-includes/include/usr/src/linux-1.2.10-includes/include/linux/usr/src/linux-1.2.10-includes/include/linux/modules/<zkráceno>#
```

Bohužel, program tar neumí zjistit změny informací v inode souboru, například změny bitu vyhra-zeného pro přístupová práva nebo změny jména souboru. Lze to obejít pomocí programu find a srovnávat stav dat uložených v souborovém systému se seznamem souborů, které byly posled-ně zálohovány. Skripty a programy, které implementují tento způsob zálohování, najdete na růz-ných linuxových FTP serverech.

Obnovování souborů programem tar

Zálohované soubory lze obnovit příkazem tar s parametrem -extract (-x):

```
# tar –extract –same-permissions –verbose –file /dev/fd0H1440 usr/src/
usr/src/linux
```

usr/src/linux-1.2.10-includes/
usr/src/linux-1.2.10-includes/include/
usr/src/linux-1.2.10-includes/include/linux/
usr/src/linux-1.2.10-includes/include/linux/hdreg.h
<zkráceno>
#

Můžete rovněž obnovovat jenom vybrané soubory či adresáře (a všechny soubory a podadresáře, které jsou v nich uloženy) tak, že je uvedete v příkazové řádce:

tar xpvf /dev/fd0H1440 usr/src/linux-1.2.10-includes/include/linux/hdreg.h usr/src/linux-1.2.10-includes/include/linux/hdreg.h

Když vás zajímá jenom to, které soubory záloha obsahuje, zadejte příkaz tar s parametrem --list (-t):

tar –list –file /dev/fd0H1440usr/src/usr/src/linuxusr/src/linux-1.2.10-includes/usr/src/linux-1.2.10-includes/include/usr/src/linux-1.2.10-includes/include/linux/usr/src/linux-1.2.10-includes/include/linux/hdreg.h<zkráceno>#

Uvědomte si, že program tar čte zálohu vždy sekvenčně, takže je při práci s většími objemy dat dost pomalý. Ale jestli používáte páskové jednotky nebo jiná média se sekvenčním přístupem, stej-ně nemůžete využít různé techniky, jež využívají přímý přístup.

Program tar neumí správně zacházet se smazanými soubory. Potřebujete-li obnovit souborový systém z úplné a inkrementální zálohy a mezi těmito zálohami jste některý ze souborů smazali, po obnovení dat ze záloh bude smazaný soubor znovu existovat. To může být dost závažný pro-blém například v případě, že soubor obsahuje citlivá data, která by již neměla být k dispozici.

Víceúrovňové zálohování

Metoda jednoduchého zálohování, jež byla načrtnuta v předchozím odstavci, je vhodná pro osob-ní potřebu nebo systémy s malým počtem uživatelů. Pro náročnější podmínky je vhodnější více-úrovňové zálohování.

Metoda jednoduchého zálohování má dvě úrovně: úplné a inkrementální zálohování. Ty lze zobecnit do libovolného počtu dalších úrovní. Úplná záloha by mohla být úrovní 0 a další násled-né inkrementální zálohy úrovněmi 1, 2, 3 a tak dále. Na každé úrovni inkrementálního zálohová-ní se zálohuje vše, co se změnilo od poslední zálohy stejné nebo nižší úrovně.

Účelem víceúrovňového zálohování je levněji prodloužit historii zálohování dat. V příkladu v před-chozím odstavci šla historie zálohování jenom k poslední úplné záloze. Kdybyste měli víc médií, mohli byste ji prodloužit, ale s každou další novou páskou jenom o týden, a to by bylo příliš nákladné. Delší historie vytváření záloh je užitečná, protože omylem smazaných nebo poškoze-ných souborů si často všimnete až po delší době. Navíc jakákoliv verze souboru, i když není zrov-na aktuální, je obvykle lepší než žádná.
Víceúrovňovým zálohováním lze historii vytváření archivů prodloužit levněji. Když si například koupíme deset pásek, můžeme používat pásky 1 a 2 na měsíční zálohy (první pátek každého měsíce), pásky 3 až 6 na týdenní zálohy (všechny ostatní pátky – uvědomte si, že v některém měsí-ci může být pět pátků, takže potřebujete o čtyři pásky víc) a pásky 7 až 10 na denní zálohy (od pondělí do čtvrtka). Takto jsme schopni – pouhým zakoupením čtyř dalších pásek navíc – pro-dloužit historii zálohování ze dvou týdnů (s využitím všech denních záloh) na dva měsíce. Prav-da, během těchto dvou měsíců nemůžeme obnovit všechny verze zálohovaných souborů, avšak to, co obnovit lze, obvykle postačí.

Z obrázku 12.1 je patrné, která úroveň zálohování se používá v ten který den a z kterých záloh je možno na konci měsíce data obnovit.

Víceúrovňové zálohování navíc snižuje i čas potřebný k obnovení celého souborového systému. Když potřebujete obnovit celý systém souborů a máte mnoho inkrementálních záloh s monotónně rostoucí řadou úrovní, musíte data pracně obnovovat postupně ze všech médií. Když ale budete pou-žívat méně záloh a větší počet úrovní, snížíte počet úrovní potřebných k obnovení celého svazku.

Chcete-li snížit počet médií potřebných k obnovení dat, používejte pro inkrementální zálohy menší rozsah úrovní. Tak se ovšem prodlouží čas zálohování, protože při každém zálohování se ukládá vše, co se změnilo od předchozí úplné zálohy. O něco lepší plán zálohování se uvádí v manuálové strán-ce programu dump. Jeho schéma je patrné z tabulky 11.1. Zkuste použít posloupnost úrovní zálo-hování: 3, 2, 5, 4, 7, 6, 9, 8, 9 a tak dále. Snížíte tím čas zálohování i obnovování dat. Nejdelší zálo-ha se bude pořizovat po dvou dnech práce. Počet pásek, z nichž se budou obnovovat data, závisí na intervalu mezi úplnými zálohami, ale je rozhodně nižší než u procedury jednoduchého zálohování.

Páska	Úroveň	Zálohuje se (dní)	Obnova z pásek
1	0	-	1
2	3	1	1, 2

3	2	2	1, 3
4	5	1	1, 2, 4
5	4	2	1, 2, 5
6	7	1	1, 2, 5, 6
7	6	2	1, 2, 5, 7
8	9	1	1, 2, 5, 7, 8
9	8	2	1, 2, 3, 7, 9
10	9	1	1, 2, 5, 7, 9, 10
11	9	1	1, 2, 5, 7, 9, 10, 11
...	9	1	1, 2, 5, 7, 9, 10, 11, ...

Efektivní zálohovací schéma při více úrovních

Tyto sofistikované postupy zálohování obvykle redukují pracnost, ale na druhou stranu je víc věcí,
které jako správce systému musíte uhlídat. Sami se musíte rozhodnout, jestli se vám to vyplatí.
Program dump má vestavěnou podporu pro úrovně zálohování. Pro tar a cpio je nutno je implementovat pomocí skriptů.

Co zálohovat

Je pochopitelné, že budete chtít zálohovat vše, co se vám na zálohovací média vejde. Výjimkou je software, jejž lze snadno obnovit z instalačních médií. Ale aplikace mohou používat mnoho růz-ných konfiguračních souborů a ty je lepší zálohovat, protože si tím ušetříte nelehkou práci, kte-rou zabere jejich opakované nastavování. Další významnou výjimkou je souborový systém /proc. Obsahuje pouze data, která jádro vytváří automaticky, a proto nemá žádný smysl je zálohovat.

Zvláště nežádoucí je zálohovat soubor /proc/kcore, protože je to pouze aktuální obraz fyzické
paměti, takže je dost velký.Nerozhodné jsou zprávy news, poštovní schránky, logovací soubory a mnoho dalších dat ulože-ných v adresáři /var. Sami se musíte rozhodnout, co pokládáte za důležité.

Typickým příkladem toho, co se musí zálohovat, jsou uživatelské soubory (adresář /home) a syste-mové konfigurační soubory (adresář /etc a další konfigurační údaje, které jsou často rozeseté po celém souborovém systému).

Komprimované zálohy

Zálohy zabírají hodně místa na disku, což může stát dost peněz. Nároky na diskový prostor lze minimalizovat komprimováním záloh. Existuje několik způsobů, jak to udělat. Některé programy přímo podporují kompresi, například po zadání parametru --gzip (-z) programu tar verze GNU se jeho výstup spojí pomocí roury s kompresním programem gzip. Záloha se pak komprimuje před tím, než se zapíše na zálohovací médium.

Bohužel, komprimované zálohy mohou způsobit vážné komplikace. Z povahy toho, jak kompri-mace funguje, vyplývá, že když se zapíše jediný bit špatně, bude nepoužitelný i celý zbytek kom-primovaných dat. Některé zálohovací programy sice používají zabudované opravné algoritmy, ale žádná z těchto metod si neumí poradit s větším počtem chyb. Je-li tedy záloha komprimovaná způsobem, jakým to dělá program tar verze GNU, který svůj výstup komprimuje jako celek, může jediná chyba způsobit poškození celé zálohy. Stěžejním rysem zálohování musí být spolehlivost, takže tato metoda komprese není příliš dobrá.

Alternativou je komprimace jednotlivých záloh (souborů). Když se pak stane, že bude některý z nich poškozen, můžete použít jinou zálohu. Pravděpodobnost, že data, která tím ztratíte, se mohou poškodit i jiným způsobem, je stejná, takže na tom nebudete o moc hůř, než kdyby se nezálohovalo vůbec. Metodu komprimace jednotlivých souborů využívá například program afio (varianta programu cpio).

Komprese nějakou dobu trvá, takže zálohovací program nemusí být schopen data dostatečně rychle zapisovat na pásku. Tento problém lze řešit využitím vyrovnávacích pamětí pro výstup zálohovacích programů (buď interních, je-li zálohovací program natolik chytrý, nebo pomocí jiných programů). Ale i tak by se mohlo stát, že zálohování nebude fungovat správně. S tímto problémem byste se měli setkat jen u velmi pomalých počítačů.

Časové údaje

Čas je iluze. Čas oběda dvojnásob. (Douglas Adams)

V této kapitole bude vysvětleno, jakým způsobem systém Linux udržuje správný čas a co je potře-ba dělat, abyste potížím s nesprávným systémovým časem předešli. Obvykle nebudete muset dělat se systémovým časem nic, ale je užitečné chápat, oč jde.

Časové zóny

Měření času je založeno na převážně pravidelných přírodních jevech, jako je střídání světla a tmy, jehož příčinou je rotace planety. Celkový čas mezi dvěma po sobě následujícími periodami je stej-ný, ale délka denní a noční doby se mění. Jedinou konstantou je poledne.

Poledne je denní doba, ve které se Slunce nachází na své nejvyšší pozici. Vzhledem k tomu, že Země je kulatá, je poledne na různých místech zeměkoule v jinou dobu. Z toho vychází předsta-va *místního času*. Lidstvo měří čas v různých jednotkách, jichž většina je rovněž svázána s pří-rodními jevy, jako je ona kulminace Slunce v poledne. Pokud se nacházíte stále na stejném místě, nezáleží na tom, že je lokální čas na jiných místech jiný.

Když ale potřebujete komunikovat se vzdálenějšími místy, uvědomíte si zároveň, že potřebujete jakýsi společný čas. V dnešní moderní době potřebuje hodně míst komunikovat s různými jinými místy na celé planetě. Proto byl zaveden společný standard měření času. Tomuto společnému času se říká *univerzální čas* (anglicky *universal time*, zkráceně UT nebo UTC), dříve označovaný jako greenwichský čas (Greenwich Mean Time nebo GMT), protože je místním časem v Greenwichi v Anglii. Když spolu potřebují komunikovat lidé, kteří mají různý lokální čas, mohou používat spo-lečný univerzální čas. Nevzniká tak zmatek kolem toho, kdy se která věc stala nebo měla stát.

Místním časům se říká časové zóny. I když se z pohledu geografie zdá logické, aby byla místa, která mají poledne ve stejnou dobu, začleněna do stejného časového pásma, neumožňují to hle-diska politická. Mnoho zemí z různých důvodů zavádí *letní čas* (anglicky *daylight savings time*, zkráceně DST). Posouvají ručičky hodinek tak, aby měly více denního světla v době, kdy se pra-cuje, pak je v zimě zase přetáčí zpět. Jiné státy to tak pro změnu nedělají. No a ty, které tak činí, nejsou zajedno v tom, kdy má být čas posunut, a mění pravidla z roku na rok. To je důvod, proč jsou konverze časů mezi pásmy poměrně netriviální.

Časová pásma je nejlepší určovat podle polohy nebo podle rozdílu mezi místním a univerzálním časem. Ve Spojených státech a některých dalších zemích mají místní časové zóny své jméno a odpovídající třípísmennou zkratku. Ale tato zkratka není jedinečná a neměla by se používat bez současného označení země. Je lepší mluvit o lokálním čase například v Helsinkách, než o východoevropském čase (zkratka EET, East European Time), protože ne všechny státy východní Evro
py leží ve stejném pásmu a nemusí mít stejná pravidla přechodu na letní čas. Linux má balík programů pro práci s časovými pásmy. Tento software zná všechny existující časo-vé zóny, a navíc jej lze jednoduše přizpůsobit v případě, že se změní pravidla určování času. Takže jediné, co musí správce systému udělat, je vybrat správné časové pásmo. Také každý z uži-vatelů si může nastavit své vlastní časové pásmo – to je důležité proto, že mnoho z nich používá prostřednictvím sítě Internet počítače v různých zemích. Když se ve vašem místním časovém pásmu změní pravidla přechodu na letní čas, budete muset aktualizovat tomu odpovídající část subsystému pro určování času. Stačí obvykle nastavit časové pásmo systému a upravit datové sou-bory zvoleného pásma, zbytek už nestojí za řeč.

Hardwarové a softwarové hodiny

Osobní počítač má hardwarové systémové hodiny napájené z baterií. Díky těmto bateriím hodiny fungují, i když je zbytek počítače bez proudu. Hardwarové systémové hodiny lze nastavovat pomocí programu setup v BIOSu nebo přímo z operačního systému.

Jádro systému Linux udržuje vlastní čas nezávisle na hardwarových hodinách. Při zavádění ope-račního systému Linux nastaví své vlastní softwarové hodiny na stejný čas, jaký je v tom momen-tě na systémových hardwarových hodinách. Pak běží oboje hodiny nezávisle. Systém Linux udr-žuje vlastní čas pomocí softwarových hodin proto, že využívání systémových hardwarových hodin je dost pomalé a složité.

Hodiny jádra systému ukazují vždy univerzální čas. Proto jádro nemusí vůbec nic vědět o časo-vých zónách – jednoduchost přispívá k vyšší spolehlivosti a ulehčuje možnost změny informací
o vybraném časovém pásmu. Každý proces si převádí časová pásma sám (pomocí standardních

nástrojů, jež jsou součástí balíku pro konverze časových zón). Hardwarové systémové hodiny mohou být nastaveny na místní či univerzální čas. Je obvykle lepší mít je nastaveny na univerzální čas, protože pak není potřeba měnit nastavení hardwarových syste-mových hodin na začátku a konci období letního času (univerzální čas je totiž „pořád zimní“). Bohužel některé operační systémy pro PC – včetně systémů MS-DOS, Windows, OS/2 – počítají s tím, že hardwarové hodiny ukazují lokální čas. Systém Linux si umí poradit s oběma způsoby nastavení, ale v případě, že hardwarové hodiny ukazují místní čas, bude potřeba měnit jejich nastavení při přechodu z a na letní čas (jinak by totiž neukazovaly místní čas).

Zobrazení a nastavování času

V Linuxu je systémové časové pásmo určeno symbolickým odkazem /etc/localtime. Tento odkaz odkazuje na datový soubor pro dané časové pásmo, popisující místní časovou zónu. Jed-notlivé datové soubory pro časová pásma jsou uloženy v adresáři /usr/lib/zoneinfo nebo /usr/share/zoneinfo, záleží na tom, jakou distribuci používáte.

Například v systému openSUSE umístěném v New Jersey bude odkaz etc/localtime ukazovat na /usr/share/zoneinfo/US/Eastern. V systému Debian bude odkaz etc/localtime ukazo-vat na /usr/lib/zoneinfo/US/Eastern.

Když se vám nepodaří najít adresář zoneinfo v žádném z adresářů /usr/lib nebo /usr/share, zadejte buď příkaz find /usr -print | grep zoneinfo nebo nahlédněte do dokumentace k pří-slušné distribuci.
Co se stane, když jsou uživatelé umístěni v různých časových pásmech? Uživatel může změnit své vlastní časové pásmo nastavením proměnné prostředí TZ. Není-li hodnota TZ nastavena, předpo-kládá se, že uživatel používá nastavení časového pásma systému. Syntaxe nastavení hodnoty pro-měnné TZ je popsána v manuálové stránce příkazu tzset.

Příkaz date ukáže aktuální datum a čas. Například:

$ date Sun Jul 14 21:53:41 EET DST 1996 $

Znamená to, že je neděle 14. července 1996, asi za deset minut deset večer, to všechno v časo-vém pásmu označeném „EET DST“, což by mohlo v angličtině znamenat „East European Daylight Savings Time“, tedy východoevropský letní čas. Příkazem date můžeme vypsat i světový čas:

$ date -u Sun Jul 14 18:53:42 UTC 1996 $

Příkazem date se také nastavují softwarové hodiny jádra systému:

date 07142157 Sun Jul 14 21:57:00 EET DST 1996 # date Sun Jul 14 21:57:02 EET DST 1996

Více podrobností hledejte v manuálové stránce příkazu date – syntaxe příkazu je tak trochu tajem-ná. Čas může nastavovat pouze superuživatel. I když může mít každý z uživatelů nastaveno své vlastní časové pásmo, systémový čas je pro všechny stejný.

Příkaz date ukazuje nebo nastavuje jenom softwarové hodiny. Příkaz clock synchronizuje syste-mové hardwarové a softwarové hodiny. Spouští se při zavádění systému, kdy se zjišťuje nastave-ní hardwarových systémových hodin. Podle nich se pak nastavují hodiny softwarové. Potřebuje-te-li nastavit oboje, nastavte nejprve softwarové hodiny příkazem date, následně hardwarové pří-kazem clock -w.

Parametrem -u sdělíte programu clock, že hardwarové hodiny ukazují světový čas. Přepínač -u je potřeba používat správně. V opačném případě bude mít váš systém mírný zmatek v tom, jaký je vlastně přesný čas.

Nastavení hodin se musí dělat opatrně. Mnoho částí operačního systému Unix spoléhá na to, že hodiny fungují správně. Například démon cron spouští pravidelně různé příkazy. Změníte-li nasta-vení hodin, může se stát, že nebude vědět, zda je potřeba některý z programů spustit či nikoliv. Když na některém starším unixovém systému někdo nastavil hodiny o dvacet let dopředu, démon cron se snažil spustit všechny periodicky vykonávané příkazy za celých dvacet let naráz. Aktuál-ní verze programu cron si s tímto problémem umí poradit. Přesto byste měli být při změnách času opatrní. Velké časové skoky a skoky vzad jsou nebezpečnější než menší změny a posuny dopře-du.

Když jdou hodiny špatně

Softwarové hodiny systému Linux nejdou vždy přesně. V chodu je udržují pravidelná *přerušení časovače* generovaná hardwarem PC. Běží-li v systému příliš mnoho procesů, může trvat obsluha přerušení časovače příliš dlouho a softwarové hodiny se začnou zpožďovat. Hardwarové syste-mové hodiny běží nezávisle na operačním systému a jsou obvykle přesnější. Jestliže často vypí-náte a zapínáte počítač (to je případ většiny systémů, které neslouží jako servery), budete mít obvykle i přesnější systémový čas.

Potřebujete-li upravit nastavení hardwarových hodin, je obyčejně nejjednodušší restartovat systém, spustit setup BIOSu a udělat to tam. Tak lze předejít všem potížím, které by mohly být zapříčině-ny změnou systémového času za běhu systému. Nemáte-li možnost upravit čas v nastaveních systému BIOS, nastavte jej příkazy date a clock (v tomto pořadí), ale připravte se na to, že se možná některé části systému budou chovat podivně, takže budete muset systém opět restartovat.

K synchronizaci hardwarových hodin podle softwarových také můžete použít buď hwclock -w nebo hwclock-systohc. Pokud byste chtěli synchronizovat softwarové hodiny podle hardwaro-vých, použijete k tomu hwclock -s nebo hwclock --hwtosys. Podrobnější informace viz manu-álové stránky hwclock.

NTP – Síový protokol pro správu času (Network Time Protocol)

Počítač na síti (i ten, který je připojený přes modem) si může automaticky kontrolovat hodiny tak, že je srovnává s časem na jiném počítači, o kterém ví, že udržuje přesný čas. K tomu slouží pro-tokol NTP. Je to způsob ověřování a seřizování času na počítači na základě synchronizace s jiným systémem. Pomocí tohoto protokolu lze v souladu s CUT (Coordinated Universal Time, koordi-novaný světový čas) udržovat správný čas s přesností na milisekundy. Další informace viz *http://www.time.gov/about.html.*

Pro běžné uživatele Linuxu je to jistě luxus. U mě doma seřizujeme všechny hodiny podle Linu-xu. Ve větších společnostech je ovšem tento „luxus“ nezbytností. Hledání v žurnálních souborech podle času každému značně usnadní život a pomůže i při ladění programů.

Jiným příkladem potřebnosti NTP je SAN. Některé SANy vyžadují správnou konfiguraci i činnost NTP kvůli synchronizaci souborových systémů a správným časovým známkám. Některé SANy i aplikace mohou být zmateny, když narazí na soubory s časovými známkami z budoucnosti.

Součástí většiny linuxových distribucí je nějaká forma NTP, buď v .deb nebo .rpm balících. Z nich nain-stalujete NTP anebo si můžete stáhnout zdrojové soubory z *http://www.ntp.org/downloads.html* a sami si je přeložit. Základní konfigurace je v obou případech stejná.

Základní konfigurace NTP

Program NTP se konfiguruje buď pomocí souboru /etc/ntp.conf nebo /etc/xntp.conf v závis-losti na konkrétní distribuci. Detaily konfigurace se nebudeme příliš zabývat, řekneme si jen to základní.

Základní soubor ntp.conf může vypadat například takto:

```
# --- GENERAL CONFIGURATION --server aaa.bbb.ccc.ddd server
127.127.1.0 fudge 127.127.1.0 stratum 10


# Drift file.


driftfile /etc/ntp/drift
```

Základní soubor ntp.conf obsahuje pouze 2 servery, jeden, s kterým se chce synchronizovat, a svoji IP pseudoadresu (v tomto případě 127.127.1.0). Pseudoadresu používá pro případ problé-mů se sítí nebo pro případ, že vzdálený NTP server bude mimo provoz. NTP se bude synchroni-zovat sám se sebou až do té doby, než se bude moci opět synchronizovat se vzdáleným serve-rem. Doporučujeme mít v souboru alespoň dva vzdálené servery, s nimiž se budete synchronizo-vat. Jeden jako primární, druhý jako zálohu.
Základní soubor by také měl obsahovat místo na uložení souboru odchylek. NTP časem odpozo
ruje velikost odchylky systému od správného času a bude ji automaticky korigovat. Lepší bezpečnost NTP a kontrolu toho, co smí NTP dělat a kdo má k němu přístup, zajistíte pomocí volby omezení. Například:

```
# Zakaž volný přístup k této službě.
restrict default ignore


# Povol systémům v této síti synchronizaci s touto časovou službou,
# avšak bez modifikace.
restrict aaa.bbb.ccc.ddd nomodify


# Povol následující neomezený přístup k ntpd


restrict aaa.bbb.ccc.ddd
restrict 127.0.0.1
```

Před zaváděním omezení doporučujeme počkat, až bude NTP pracovat, jak má. Omylem byste

mohli omezit sami sebe a zbytečně strávit čas laděním, proč tomu tak je.NTP neopravuje systémový čas hned. Musíte být trpěliví. NTP jednoduše otestujete tak, že 10minut před tím, než půjdete spát, posunete systémový čas o 10 minut a po probuzení jej zkontrolujete. Měl by být opravený.
Mnozí uživatelé si myslí, že místo démona NTP stačí pouze nastavit úlohu cron tak, aby pravidelně spouštěla příkaz ntpdate. Tato metoda má dvě podstatné nevýhody.První je, že ntpdate mění čas „brutální silou“. Je-li tedy čas posunutý o 5 minut, opraví jej ihned.V některých prostředích může taková náhlá změna času způsobit problémy – například když pou-žíváte bezpečnostní software citlivý na čas, můžete někomu nechtěně zrušit přístup k systému.Aby se NTP vyhnul takovým nepříjemnostem, mění čas pomalu.

Druhým důvodem je skutečnost, že démon NTP lze zkonfigurovat tak, aby se pokusil zjistit odchylku od správného času a automaticky ji korigoval.

Nástroje NTP

Pro kontrolu NTP existuje mnoho různých nástrojů. Stav běžného systémového času můžete vypsat například pomocí příkazu ntpq -p.

```
# ntpq -p
      remote refid st t when poll reach  delay offset jitter
============================================================================== *cudns.cit.corne ntp0.usno.navy.
2 u 832 1024 377 43.208 0.361 2.646
                                    LOCAL(0) LOCAL(0) 10 l  13 64 377 0.000 0.000 0.008
```

Příkazem ntpdc -c loopinfo si vypíšete, jaká je odchylka systémového času v sekundách v závis-losti na posledním kontaktu se vzdáleným serverem.

```
# ntpdc -c loopinfo
offset:              -0.004479 s
frequency:           133.625 ppm
poll adjust:         30
watchdog timer:      404 s
```

Příkaz ntpdc -c kerninfo vypíše, kolik zbývá ke zkorigování.

```
# ntpdc -c kerninfo pll offset: -0.003917 s pll frequency: 133.625 ppm maximum error: 0.391414 s estimated error: 0.003676 s status: 0001 pll pll time constant: 6 precision: 1e-06 s frequency tolerance: 512 ppm <zkráceno>
```

Poněkud odlišná verze ntpdc -c kerninfo je ntptime.

```
# ntptime
ntp_gettime() returns code 0 (OK) time c35e2cc7.879ba000 Thu, Nov 13 2003 11:16:07.529, (.529718), maximum error 425206 us, estimated error 3676 us
ntp_adjtime() returns code 0 (OK) modes 0x0 (), offset -3854.000 us, frequency 133.625 ppm, interval 4 s, maximum error 425206 us, estimated error 3676 us, status 0x1 (PLL), time constant 6, precision 1.000 us, tolerance 512 ppm, pps frequency 0.000 ppm, stability 512.000 ppm, jitter 200.000 us, intervals 0, jitter exceeded 0, stability exceeded 0, errors 0.
```

Další možností, jak zjistit, jak pracuje NTP, je příkaz ntpdate -d. Příkaz určí časový rozdíl na zákla-dě kontaktu s NTP serverem, avšak systémový čas nezmění.

```
# ntpdate -d 132.236.56.25013 Nov 14:43:17 ntpdate[29631]: ntpdate 4.1.1c-rc1@1.836 Thu Feb 13 12:17:20EST 2003
(1)transmit(132.236.56.250)receive(132.236.56.250)transmit(132.236.56.250)receive(132.236.56.250)transmit(132.236.56.250)receive(132.236.56.250)transmit(132.236.56.250)receive(132.236.56.250)transmit(132.236.56.250)server 132.236.56.250, port 123stratum 2, precision -17, leap 00, trust 000refid [192.5.41.209], delay 0.06372, dispersion 0.00044transmitted 4, in filter 4<zkráceno>
13 Nov 14:43:17 ntpdate[29631]: adjust time server 132.236.56.250 offset 0.001020 sec
```

Chcete-li sledovat synchronizaci systému, můžete zadat ntptrace.

```
# ntptrace 132.236.56.250 cudns.cit.cornell.edu: stratum 2, offset -0.003278, synch distance 0.02779 truetime.ntp.com: stratum 1, offset -0.014363, synch distance 0.00000, refid 'ACTS'
```

Pokud potřebujete systémový čas synchronizovat okamžitě, můžete to provést příkazem ntpdate remote-servername. Bez čekání.

```
# ntpdate 132.236.56.250 13 Nov 14:56:28 ntpdate[29676]: adjust time server 132.236.56.250 offset 0.003151 sec
```

Některé známé NTP servery

Seznam veřejných NTP serverů lze získat na *http://www.eecis.udel.edu/~mills/ntp/servers.html.* Pro-sím, přečtěte si nejdříve pečlivě návod, jak se jednotlivé servery používají. Ne všechny mají vyso-korychlostní přístup a nemusí zvládat velký počet synchronizací. Než začneme využívat služby NTP serveru, je vhodné nejprve kontaktovat jeho správce.

Odkazy na NTP

Podrobnější informace o NTP naleznete na domovské stránce NTP: *http://www.ntp.org* nebo na *http://www.ntp.org/ntpfaq/NTP-a-faq.htm.*

O nápovědě

„Help me if you can I'm feeling down. And I do appreciate you being 'round.“

-The Beatles

Nápověda existuje. Jenom musíte vědět, kde ji hledat. V Linuxu je takových míst opravdu hodně. Rozesílací seznamy, kanály IRC, www stránky s veřejnými fóry a mnoho dalších zdrojů. V této kapitole se vám pokusíme napovědět, kde hledat nápovědu.

Diskusní skupiny a e-mailové konference

Tento průvodce vás nemůže naučit o Linuxu úplně všechno, neboť zde na to není dost místa. Je zcela jisté, že v určitém okamžiku zjistíte, že potřebujete provést něco, co není popsáno v žádném dokumentu LDP.

Jedna z nejhezčích věcí na Linuxu je, že existuje tolik míst, která se mu věnují. Prakticky ke kaž-dému linuxovému tématu existuje nějaké fórum, a to od začátečníků až po nejsložitější problémy jádra. Zvolíte-li správný postup, není problém se k většině z nich dostat.

Jak najít správné fórum

První věc, kterou musíte udělat, je najít odpovídající fórum. Existuje mnoho diskusních skupin a rozesílacích seznamů věnovaných Linuxu. Zkuste nalézt takový, který je nejbližší vašemu pro-blému. Není například příliš vhodné se ptát na sendmail na fóru věnovaném jádru Linuxu. Účast-níci si přinejmenším pomyslí něco o vaší hlouposti a může se stát, že dostanete řadu velmi nesluš-ných odpovědí. Přitom během okamžiku lze v diskusních skupinách najít skupinu comp.mail.sendmail, která by mohla být vhodným místem pro kladení otázek o sendmailu. Váš zpravodajský klient pravděpodobně má seznam diskusních skupin. Pokud ne, jejich kompletní seznam naleznete na *http://groups.google.com/groups?group=**.

Než napíšete

Nyní, když jste nalezli správné fórum, můžete se domnívat, že nic nebrání odeslání otázky. Stop, ještě chvilku počkejte. Pokusili jste se už sami nalézt odpověď na vaši otázku? Existuje nepřeber-né množství návodů a často kladených otázek, a pokud se některé z nich vztahují k vašemu pro-blému, *přečtěte si je nejdříve*. Ani když na vaši otázku neodpoví, dozvíte se alespoň něco o téma-tech a vaše otázka bude zasvěcenější a přesnější. Také existují archivy diskusních skupin a roze-sílacích seznamů a je docela možné, že už někdo dříve položil stejnou otázku a někdo na ni odpo-věděl. *Než* položíte otázku, zkuste *http://www.google.com* nebo některý podobný vyhledávač.

Co napsat

Dobrá, našli jste si tedy odpovídající fórum, přečetli jste si správné návody a často kladené otáz-ky, prohledali jste Internet, ale odpověď na vaši otázku jste nikde nenašli. Nyní můžete začít psát svoji otázku. Není špatné se nejprve zmínit, že o daném tématu jste si už něco přečetli. Můžete napsat třeba: „Přečetl jsem si návod k Winmodemu a prošel často kladené otázky, nikde jsem však nenašel nic o „Nastavení Winmodemu v Linuxu“ a ani Google mně nic nenašel.“ Budete aspoň vypadat, jako že jste projevili

trochu snahy, a ne jako líný hlupák, který se sám neumí pomalu ani najíst. Zná-li někdo odpověď, spíš vám pomůže a neriskujete nezájem nebo rovnou posměšky.

Pište jasně, správnou angličtinou a bez pravopisných chyb. To je mimořádně důležité. Projevíte se jako člověk, který myslí přesně a uvážlivě. Nepište „u“ místo „you“ a „b4“ místo „before“. Snažte se působit jako vzdělaný a inteligentní člověk, a ne jako idiot. Pomůže to, slibuji.
Také nepište velkými písmeny, TAKTO. Je to, jako kdybyste křičeli, a vypadá to neotesaně. Popište podrobně, v čem spočívá problém a jak jste se jej pokusili odstranit. Otázka jako „Nefun-guje mně Linux, co mám dělat?“ je úplně k ničemu. Kde nefunguje? Jak nefunguje? Musíte být co
nejpřesnější, ale zase to nepřehánějte. Vynechte nepodstatné informace. Máte-li problém s poš-tovním klientem, je asi zbytečné přikládat k otázce výpis zavaděče jádra (dmesg). Nevyžadujte odpověď na svoji soukromou e-mailovou adresu. Kouzlo linuxových fór spočívá
v tom, že účastníci se učí jeden od druhého. Žádostmi o soukromou odpověď diskusní skupiny a konference znehodnocujete.

V jakém tvaru

Nepište v HTML. Mnozí uživatelé Linuxu mají poštovní klienty, kteří nedokážou jednoduše přečíst zprávu v HTML. Museli by vyvinout nějakou snahu a pak by HTML přečetli, ale nechce se jim. HTML e-maily se občas zahazují. Pište v prostém textu a budete mít více čtenářů.

Hodnocení

Až naleznete řešení, proveďte krátké zhodnocení a vysvětlete, v čem byl problém a jak jste jej odstranili. Uživatelé to ocení a nejen že se celá záležitost uzavře, ale určitě v budoucnosti něko-mu pomůžete s podobným problémem. Při prohledávání archivu diskusních skupin nebo rozesí-lacích seznamů uvidí, že už někdo podobný problém řešil, projde si následnou diskusi a najde konečné řešení.

Další informace

Tento stručný průvodce je parafrází a shrnutím vynikajícího (a podrobnějšího) dokumentu „Jak klást chytré otázky“ od Erika S. Raymonda (na adrese *http://www.catb.org/~esr/faqs/smart -questions.html*). Doporučujeme si jej přečíst, než něco pošlete. Dokážete lépe zformulovat otáz-ku a zvýšíte pravděpodobnost toho, že dostanete odpověď, kterou hledáte.

IRC

IRC (Internet Relay Chat) není součástí dokumentu Erika Raymonda, avšak je to také vynikající způsob, jak nalézt to, co potřebujeme vědět. Tato metoda však vyžaduje určitou praxi v pokládá-ní správných otázek. Většina sítí IRC má linuxové kanály zaplněné, a je-li odpověď na vaši otáz-ku možné nalézt v manuálových stránkách nebo v návodech, má se zato, že ji budete hledat tam. Pravidlo o správné angličtině platí i v tomto případě.
Většina toho, co bylo řečeno o diskusních skupinách a konferencích, platí i pro IRC s následujícími dodatky.

Barvy

Nepoužívejte barvy, tučné písmo, kurzivu ani neznámé znaky (jež nepatří do ASCII). Starší terminály by si s tím nevěděly rady a vypadá to škaredě. Když se napojíte na kanál a začnete chrlit barvy nebo tučné znaky, určitě vás brzy vyhodí.

Bute slušní

Uvědomte si, že na odpověď nemáte nárok. Zeptáte-li se správně, pravděpodobně odpověď dostanete, ale nárok na ni nemáte. Lidé kolem IRC využívají kanál ve svém volném čase a nikdo jim za to neplatí, zvláště vy ne.

Buďte slušní. Chovejte se k ostatním tak, jak byste chtěli, aby se oni chovali k vám. Domníváte-li se, že se někdo nechová slušně, nezačněte mu hned nadávat a nerozčilujte se – chovejte se ještě slušněji. Tím z něho spíš uděláte hlupáka, než kdybyste se snížili na jeho úroveň.

Nepokoušejte se o žargon jako „liskat někoho velkým pstruhem přes hubu“ a podobně. Věřili byste, že už to zkoušel někdo před vámi? Ani napoprvé to nebylo moc vtipné.

Pište správně anglicky

Většina #linuxových kanálů je v angličtině. Tam pište anglicky. Většina větších IRC sítí má také #linuxové kanály v jiných jazycích, např. francouzský je #linuxfr, španělský #linuxes nebo #linux-latino. Nemůžete-li najít správný kanál, určitě vám pomůže někdo z anglického kanálu #linux.

Nepište takové nesmysly jako „1337 H4X0R d00d!!!“, ani když jiní to tak dělají. Vypadá to hloupě a hloupě budete vypadat i vy. Přinejlepším si o vás ostatní pomyslí, že jste idiot, ale spíš se vám vysmějí a pak vás vyhodí.

Skenování portů

Nikdy po nikom nechtějte, aby skenoval vaše porty nebo aby vás zkusil „hacknout“. To je zákon. Neexistuje způsob, jak se dozvědět, že jste tím, za koho se vydáváte, nebo že IP adresa, z níž jste připojen, patří vám. Nevystavujte nikoho do situace, že by vám takový požadavek musel odmít-nout.

Nikdy nikomu neskenujte porty, ani kdyby vás o to požádal. Nemáte možnost o nikom zjistit, že je tím, za koho se vydává, nebo že IP adresa, z níž je připojen, patří jemu. V některých zemích je skenování portů nezákonné a v žádném případě není v souladu s podmínkami, které jste si dohodli s poskytovatelem připojení k Internetu. Většina uživatelů se přihlašuje protokolem TCP a mohou zjistit, že jsou skenováni. V takovém případě by vás určitě oznámili vašemu poskytova-teli. Zjistit, kdo skenuje, je triviální záležitost.

Vše ponechte v kanálu

Nikomu neposílejte zprávu, dokud vás o to nepožádá. Je to na úkor užitečnosti kanálu a někteří uživatelé to úplně odmítají.

Držte se tématu

Držte se tématu. Je to linuxový kanál, nikoli kanál „co dělal strýček o víkendu“. Ani když zjistíte, že někdo je mimo téma, neznamená to, že jej budete následovat. Mohou to být provozovatelé kanálu a na ně se vztahují jiná pravidla.

CTCP

Pokud uvažujete o použití protokolu CTCP na sledování kanálu nebo verzí nebo čehokoli jiného,

raději si to rozmyslete. Nejspíš by vás brzy vyhodili.Pro ty, kdo příliš neznají IRC, CTCP je protokol „Client To Client Protocol“ a je to metoda, jak zís-kávat informace o jiných klientech. Podrobnosti viz dokumentace k IRC.

Hacking, Cracking, Phreaking, Warezing

Pokud nechcete zažít odchod bez pocty, neuvažujte o napadení kanálu.Nepokoušejte se být v kanálech hackerů, crackerů, phreakerů a warezáků, když jste v #linuxovémkanále. Zdá se, že ti, kteří pracují pro #linuxové kanály, z nějakého důvodu nenávidí osoby, kteréjiným ničí počítače nebo kradou software. Ani nevím proč.

Závěrem

Omlouvám se, pokud toto povídání vypadá spíš jako hromada zákazů a málo návodů. Myslím, že

těch návodů bylo dost už v částech o diskusních skupinách a konferencích.Pravděpodobně nejlépe uděláte, když se posadíte k nějakému #linuxovému kanálu a budete jejřekněme půl hodiny sledovat. Pomůže vám to nalézt ten správný tón do diskuse.

Další texty

Existuje mnoho vynikajících míst s často kladenými otázkami, na nichž se dozvíte, jak se dostat do #linuxových kanálů. Většina jich také má část vyhrazenou často kladeným otázkám a pravi-dlům pro používání kanálu. V předmětu kanálu (který si můžete vypsat pomocí příkazu /topic) bývá obvykle uvedeno, jak je najdete. Pravidla si určitě přečtěte a dodržujte je. Jedna taková obec-ně použitelná pravidla a rady naleznete pod názvem „Undernet #linux FAQ“ na adrese *http://linuxfaq.quartz.net.nz.*

Slovníček

obecnému porozumění tak, že napíše orangutansko-lidský slovník.
Pracoval na něm celé tři měsíce. Nebylo to lehké.
Zatím se dostal jenom ke slůvku „oook“.

(Terry Pratchett, Muži ve zbrani)

Zde je prozatím stručný (doufejme, že ne nadlouho) seznam slovních definicí pojmů spojovaných s operačním systémem Linux, zvláště pak se správou systému.

account

Viz účet.

aplikační program

Software, který dělá něco užitečného. Výsledek používání aplikačního programu je v podstatě důvodem zakoupení počítače. Viz také systémový program, operační systém.

bad block

Viz vadný blok.

bad sector

Viz vadný sektor.

boot sektor

Typicky první sektor diskového oddílu. Obsahuje velmi krátký program (řádově stovky bajtů), který zajistí nahrání a spuštění operačního systému.

bootování

Cokoliv, co se děje od okamžiku zapnutí počítače po okamžik, kdy je počítač plně připraven k práci.

bootstrap loader

Velmi krátký program (umístěný typicky v paměti RAM), který přečte určitou oblast disku a předá jí řízení. Data v této oblasti uložená jsou obvykle trochu delší a složitější a dokážou zavést a spustit operační systém.

CMOS RAM

CMOS znamená „Complementary Metal Oxide Semiconductor“. Jde o složitou technologii, výsledkem je ale tranzistor, který si svůj stav pamatuje i po odpojení napájení, takže vzniká určitý typ statické paměti RAM. Tato paměť neztratí svůj obsah po vypnutí počítače.

cylindr

Množina stop na vícehlavovém disku, k níž mají hlavy přístup bez vystavování. Jinými slovy jsou to stopy, které mají stejnou vzdálenost od hřídele, kolem níž disky rotují. Když na cylin-dr umístíme data, která se zapisují nebo čtou ve stejném okamžiku, významně se sníží doba odezvy, neboť vystavování čtecí a zápisové hlavy je ve srovnání s rotací disku mnohem poma-lejší.

daylight saving time

Letní čas.

démon

Proces číhající v pozadí – obvykle nenápadně – až do chvíle, než jej něco „rozjede“. Například démon update se probudí každých třicet sekund (nebo tak nějak) a vyprázdní buffer vyrov-návací paměti, démon sendmail se probere pokaždé, když někdo odesílá poštu.

emergency boot floppy

Viz záchranná disketa.

fibre channel

Vysokorychlostní síťový protokol používaný zejména v SAN (Storage Area Networks). Navzdo-ry názvu může být realizován nejen optickými kabely, nýbrž i klasickými měděnými vodiči.

file system (souborový systém) Pojem, který má dvě použití a dva poněkud rozdílné významy. Je to jednak souhrn všech sou-borů a adresářů na mechanice (může to být pevný disk, disketa, CD-ROM atd.). Anebo jsou to značky zapsané na disk, jimiž se řídí operační systém při zápisu souborů (inody, bloky, superbloky atd.). Skutečný význam lze prakticky vždy odvodit ze souvislosti.

formátování

Přísně vzato rozdělení povrchu disku na stopy, sektory a cylindry. Někdy se (nesprávně) za součást formátování považuje i operace vytvoření souborového systému na naformátovaném médiu (zejména ve světě MS Windows/MS DOS).

fragmentace

Situace, kdy soubor není na disku zapsán v za sebou jdoucích blocích. Pokud na disku není dostatek souvislého místa k zapsání celého souboru, může být soubor zapsán ve více celcích. Tento jev se označuje jako fragmentace a vede ke zpomalenému přístupu k souboru.

full backup (úplná záloha) Kopie celého souborového systému na zálohovacím médiu (pásce, disketě nebo CD).

geometrie (disku)

Kolik má disk povrchů, cylindrů a sektorů.

high level formatting (formátování vyšší úrovně) Nesprávný pojem pro zápis souborového systému na disk. Používá se často ve světě MS Win-dows a MS-DOS.

inkrementální záloha

Zálohovací metoda, při níž se na zálohovací médium zapíší pouze data změněná od poslední úplné zálohy.

inode

Datová struktura obsahující informace o souboru. Každý soubor má svůj inode, soubor je jed-noznačně identifikován svým souborovým systémem a číslem svého inode. Každý inode obsa-huje následující informace: zařízení, na němž se inode nachází, informace o zamykání, mód a typ souboru, počet odkazů na soubor, identifikátory vlastníka a skupiny souboru, délku sou-boru, časy přístupu a modifikace, čas modifikace inode a adresy bloků, v nichž je soubor ulo-žen. Asociaci mezi soubory a čísly inode zajišťují adresáře. Inode souboru lze získat přepína-čem -i příkazu ls.

jádro systému

Část operačního systému, která implementuje interakce s technickými prostředky výpočetního systému a sdílení zdrojů. Viz také systémový program.

kořenový souborový systém

Rodič všech souborových systémů v souborovém stromu Unixu. Pokud se nepodaří tento systém připojit, jádro zpanikaří a systém se nezavede.

logický diskový oddíl

Diskový oddíl vytvořený na rozšířeném oddílu, neexistuje fyzicky, pouze v logických softwarových strukturách.

lokální čas

Čas v dané lokalitě, odpovídající otáčení Země kolem osy.

logical partition (logická oblast) Oblast v rozšířené oblasti, která je „logická“, tj. neexistuje reálně, nýbrž jen v logické softwa-rové struktuře.

logical volume manager (správce logických svazků, LVM) Nástroje, jimiž se větší fyzický disk rozdělí na „logické“ disky, které se mohou zmenšovat nebo zvětšovat v závislosti na objemu ukládaných dat.

low level formatting (formátování nižší úrovně) Synonymum pro *formátování* používané ve světě MS-DOS, jež se liší od vytváření souboro-vého systému, které se také někdy nazývá formátování.

MBR – Master Boot Record

(Obvykle) první sektor na disku, kde BIOS hledá zavaděč operačního systému.

MTA – Mail Transfer Agent

Program zodpovídající za doručování e-mailových zpráv. Po přijetí zprávy od MTU nebo jiného MTA ji dočasně uloží, analyzuje adresáta a buď zajistí její lokální doručení, nebo ji předá jinému MTA. V obou případech může měnit a doplňovat hlavičky zprávy. Nejrozšířenějším MTA v Unixu je sendmail.

MUA – Mail User Agent

Program umožňující uživateli vytvářet a číst zprávy elektronické pošty. Představuje rozhraní mezi uživatelem a mezi MTA. Odchozí zpráva je předaná prostřednictvím MTA k doručení. Pří-chozí zprávy jsou převzaty z místa, kde je zanechal MTA (i když MUA běží i na jednouživa-telských počítačích, může přebírat poštu pomocí POP). MUA jsou například pine, elm a mutt.

NFS – Network File System

Protokol vyvinutý společností Sun, popsaný v RFC 1094, který umožňuje přístup k souborům přes síť stejně, jako by byly uloženy lokálně.

operační systém

Software, jenž umožňuje sdílení zdrojů počítačového systému (procesor, paměť, diskový pro-stor, síť a tak dál) mezi uživateli a aplikačními programy, které uživatelé spouští. Nabízí zabez-pečení systému řízením přístupu k němu. Viz také jádro systému, systémový program, apli-kační program.

partition (oblast) Logická část disku. Každá oblast má obvykle svůj souborový systém. Unix má sklon k tomu, považovat oblasti za samostatné fyzické jednotky. password file (soubor s hesly) Soubor, v němž jsou uložena uživatelská jména a informace o účtech uživatelů, například hesla. V systémech typu Unix se soubor obvykle jmenuje /etc/passwd. Ve většině moderních Linuxů nejsou hesla uložena v /etc/passwd. Z bezpečnostních důvodů bývají uložena v sou-borech /etc/shadow. Další informace viz manuálové stránky passwd(5) a shadow(5). physical extents (fyzické rozšíření) Pojem používaný k popisu části fyzického svazku. Jeho rozdělení provádí manažer logických svazků (LVM) physical volume (fyzický svazek) Pojem obvykle používaný v manažeru logických svazků (LVM). platters (fyzické disky) Fyzický disk (záznamové médium) v pevné mechanice. Obvykle je pevná mechanika vytvo-řena z několika záznamových médií srovnaných nad sebou. V češtině se používá výraz plot-na. POST – Power On Self-Test

Série diagnostických testů prováděná po zapnutí počítače. Zahrnuje testy paměti, testy hardwaru, testy zapsané konfigurace a podobně.

řadič disku

Hardwarové zařízení, překládající příkazy operačního systému přímo elektronice disku. Představuje abstrakční vrstvu, díky níž operační systém nemusí umět pracovat se všemi možnými typy disků, ale pouze s omezenou množinou řadičů. Typické řadiče jsou IDE, SATA a SCSI.

souborový systém

Metody a datové struktury používané operačním systémem k udržování přehledu o souborech na disku nebo oddílu, též způsob organizace disku.

partition (oblast) Logická část disku. Každá oblast má obvykle svůj vlastní souborový systém. Unix má sklon k tomu považovat oblasti za samostatné fyzické jednotky.

sectors (sektory) Nejmenší část stopy, do níž mohou být uložena data. Obvykle (ne však vždy) má délku 512 bajtů. shadow passwords (stínová hesla) Vzhledem k tomu, že soubor s hesly v Unixu musí být často přístupný všem, obvykle neob-sahuje zašifrovaná hesla k uživatelským účtům. Místo toho se využívá stínový soubor (který nemohou číst „ostatní“), v němž jsou uložena zašifrovaná hesla k uživatelským účtům.

single user mode (jednouživatelský režim) Obvykle je definován jako úroveň běhu 1 (runlevel 1), což je úroveň, kdy není povoleno při-hlašování jiných uživatelů než root. Používá se při opravách systému (je-li souborový systém částečně poškozen, avšak je možno jej zavést v úrovni 1 a opravit jej) nebo při přesunech sou-borových systémů mezi oblastmi. To jsou jen dva příklady. Jinak ale každá úloha, která vyža-duje, aby mohl na disk v daném okamžiku psát pouze jeden uživatel, je kandidátem na úro-veň 1.

spool (řazení do fronty) Posílání souboru (nebo dat) do fronty. Obvykle se používá pro tisky, využití je však možné i jinde (například v poště). Pojem je údajně zkratkou (akronymem) vytvořenou ze slov „Simul-taneous Peripheral Operation On-Line“, avšak podle Jargon File (*http://catb.org/jargon/*) je to spíš „backronym“, tedy efektní akronym vytvořený dodatečně.

system call (systémové volání) Služba, kterou poskytuje aplikačním programům jádro, a způsob jejího vyvolání. Viz sekci 2 manuálových stránek.

swap space (odkládací prostor) Prostor na disku, do něhož systém ukládá část paměti. Obvykle je to oblast určená pro tyto účely, může to však být i soubor.

system program (systémový program) Program, který implementuje funkcionalitu na nejvyšší úrovni operačního systému, tj. činnos-ti, které nezávisí přímo na hardwaru. Někdy mohou mít zvláštní přístupová práva (např. při doručování elektronické pošty), avšak obvykle jsou považovány za součást systému (např. pře-kladač). Viz také aplikační program, jádro a operační systém.

time drift (časová odchylka) Výraz související s nepřesností počítače a udržováním přesného času. Každý počítač má při měření času určitou odchylku. V nových počítačích je míra této chyby zcela nepatrná.

track (stopa) Část *povrchu disku*, která ubíhá pod *čtecí a zapisovací hlavou*. Hlava je pevná, zatímco disk se otáčí. Stopy jsou rozděleny na *sektory*, stopy ležící nad sebou se nazývají *cylindr*.

účet

V Unixu mají uživatelé své *účty*. Ty představují jméno a heslo, jimiž se hlásí do systému, domovský adresář pro ukládání souborů, přístupová práva k hardwaru a softwaru – vše dohro-mady to tvoří účet.

úplná záloha

Zkopírování celého obsahu zálohovaného média na zálohovací médium.

souborový systém

Metody a datové struktury, které používá operační systém pro ukládání záznamů souborů na disk či diskovou oblast; způsob, kterým jsou soubory na disku organizovány. Pojem se pou-žívá též pro označení diskové oblasti či disku, kam se soubory ukládají, případně pro určení typu souborového systému.

systémový program

Programy, které implementují vyšší úrovně funkcionality operačního systému, tedy ty vlastnosti systému, které nejsou přímo závislé na hardwaru. Někdy mohou vyžadovat ke spuštění zvláštní oprávnění (například doručování elektronické pošty), ale velmi často jsou považovány za běžnou součást systému (například překladač). Viz také aplikační program, jádro systému, operační systém.

tisková fronta

Soubor nebo soubory, do nichž tiskový démon ukládá úlohy, které se mají vytisknout.

vadný blok

Blok disku (obvykle sektor), na němž nelze spolehlivě uložit data.

vadný sektor

Prakticky totéž jako vadný blok, pouze o něco přesnější určení v případě, že velikost sektoru a bloku jsou různé.

volání systému

Služby, které poskytuje aplikačním programům jádro systému, a způsob, jímž jsou volány. Viz sekci 2 manuálových stránek.

záchranná disketa

Disketa, z níž je systém možné spustit i v případě poškození disku. Většina distribucí umožňuje vytvořit tuto disketu při instalaci, což vřele doporučujeme.

ČÁST III

Bash pro začátečníky
Úvod

Proč tato příručka?

Hlavním důvodem pro vznik tohoto dokumentu bylo to, že většina čtenářů shledává existující HOWTO (*http://tldp.org/HOWTO/Bash-Prog-Intro-HOWTO.html*) za příliš krátké a neúplné, zatím-co příručka Advanced Bash Scripting (*http://tldp.org/LDP/abs/html/*) je příliš referenční. Mezi těmi-to dvěma extrémy nic jiného neexistovalo. Druhým důvodem pro vznik této příručky bylo obec-né přesvědčení, že není k dispozici dostatek volně dostupných učebních textů.

Příručka je orientována prakticky, a i když není vždy úplně vážná, snaží si namísto teoretických příkladů vybírat příklady ze života. Důvodem je zejména to, že mě neuchvacují ořezané a zjed-nodušené příklady uváděné lidmi, kteří problematice skvěle rozumějí a demonstrují některé skvě-lé funkce bashe natolik vytržené z kontextu, že je stejně nikdy nebudete moci prakticky použít. Těmito příklady se můžete zabývat až po přečtení této příručky, která obsahuje cvičení a příkla-dy, jež vám mají pomoci přežít v normálním světě.

Z vlastních zkušeností uživatele Unixu/Linuxu, administrátora a školitele vím, že lidé za sebou mohou mít roky denní práce se systémem, aniž by měli tušení o zautomatizování běžných úkonů. Proto si často myslí, že Unix není uživatelsky příjemný, a co je ještě horší, mají dojem, že je poma-lý a staromódní. To je další problém, který se tato příručka snaží řešit.

Kdo by měl příručku číst?

Kdokoliv, kdo pracuje v Unixu nebo unixovém systému chce si usnadnit život. Příručka je vhod-ná jak pro pokročilé uživatele, tak pro správce. Uživatelé, kteří mají zkušenosti s prací se systé-mem v příkazovém řádku, se dozvědí o výhodách a nevýhodách skriptů, díky nimž si mohou kaž-dodenní práci usnadnit. Práce správce systému je často na skriptech závislá, mnoho běžných úkonů se automatizuje právě pomocí skriptů. Příručka obsahuje mnoho příkladů, které vám usnadní tvorbu vlastních skriptů a nabídnou vám inspiraci pro vylepšení existujících skriptů.

Požadované znalosti (nejsou zde popisovány):

Zkušenost s užíváním Unixu či Linuxu, seznámení se základními příkazy, manuálovými stránkami a dokumentací
Znalost práce s textovým editorem
Znalost procesů spouštění a zastavování systému, inicializačních skriptů
Vytváření uživatelů a skupin, nastavování hesel
Přístupová práva, speciální práva
Znalost konvencí pro pojmenovávání zařízení a oddílů, připojování a odpojování souborových systémů.
Přidávání a odebírání programů.

Pokud některá z uvedených témat neznáte, doporučujeme první část knihy – „Úvod do systému Linux“. Další informace naleznete v dokumentaci a na jiných místech LDP (*http://www.tldp.org*).

Co budete potřebovat?

bash, *http://www.gnu.org/directory/GNU.* Bash najdete na téměř každém linuxovém systému i na celé řadě unixových systémů.
Pokud si potřebujete přeložit vlastní, rovněž by to neměl být problém – překlad je testován na mnoha Unixech, Linuxech, MS Windows i na dalších systémech.

Členění příručky

V této příručce popisujeme postupy užitečné při každodenní práci zkušenějšího uživatele. Potře-bujete sice základní znalosti o používání shellu, v prvních třech kapitolách se ale budeme věno-vat popisu základních komponent shellu a obvyklých postupů.

Kapitoly čtyři až šest popisují základní nástroje, které se ve skriptech často používají.

V kapitolách osm až dvanáct hovoříme o konstrukcích, které se ve skriptech používají.

Každá kapitola končí cvičeními, na nichž si můžete ověřit připravenost na další kapitolu.

Kapitola 1: Základy bashe – proč je bash tak dobrý, základní stavební bloky, první návody na tvorbu dobrých skriptů.
Kapitola 2: Základy skriptů – tvorba a ladění.
Kapitola 3: Prostředí bashe – inicializační soubory, proměnné, uvozování znaků, pořadí expanze, aliasy, volby.
Kapitola 4: Regulární výrazy – úvod.
Kapitola 5: Sed – úvod k řádkovému editoru Sed.
Kapitola 6: Awk – úvod do programovacího jazyka Awk.
Kapitola 7: Podmíněné příkazy – konstrukce používané k testování podmínek.
Kapitola 8: Interaktivní skripty – tvorba uživatelsky příjemnějších skriptů, uživatelský vstup.
Kapitola 9: Opakované provádění příkazů – nástroje pro tvorbu smyček.
Kapitola 10: Více o proměnných – definice typu proměnné, práce s poli, operace nad proměnnými.
Kapitola 11: Funkce – úvod.
Kapitola 12: Zachycování signálů – úvod do zpracování signálů, zachycování uživatelem posílaných signálů.

Bash a skripty v bashi

V této úvodní kapitole:

Popíšeme některé běžné shelly
Zvýrazníme výhody a možnosti GNU bashe
Popíšeme základní stavební kameny bashe
Budeme hovořit o inicializačních souborech bashe
Uvidíme, jak shell provádí příkazy
Podíváme se na několik jednoduchých příkladů skriptů

Některé běžné shelly

Obecná funkce shellu

Unixový shell interpretuje uživatelské příkazy, ať už zadávané přímo uživatelem anebo načítané ze souboru, kterému říkáme shellový skript. Skripty jsou interpretovány, nekompilují se. Shell čte příkazy ze skriptu řádek po řádku a hledá tyto příkazy v systému (viz kapitolu „Výhody Bourne Again Shellu“), zatímco kompilátor převede program do procesorem čitelné podoby – spustitelného souboru (který pak můžeme volat ve skriptu).

Kromě předávání příkazů jádru je hlavním úkolem shellu poskytnutí uživatelského rozhraní, které je možno individuálně nastavit prostřednictvím konfiguračních souborů.

Typy shellů

Stejně jako lidé používají různé jazyky a dialekty, unixový systém typicky nabízí různé shelly:

sh nebo Bourne Shell: původní shell dodnes používaný na unixových systémech a jejich derivátech. Jde o základní shell, malý program s nemnoha funkcemi. I když to není stan-dardně používaný shell, stále jej najdete na všech linuxových systémech kvůli kompatibi-litě s unixovými programy.
bash nebo Bourne Again Shell: standardní GNU shell, intuitivní a flexibilní. Pravděpodob-ně nejvhodnější pro začínajícího uživatele, zároveň však dostatečně mocný pro potřeby pokročilých a profesionálních uživatelů. Na Linuxu je bash standardním shellem běžných uživatelů. Jde o takzvanou *nadmnožinu* Bourne shellu, sadu doplňků a modulů. Zname-ná to, že Bourne Again Shell je zpětně kompatibilní s Bourne Shellem – příkazy, které fun-

gují v sh, budou fungovat i v bashi. Opačně to ne vždy platí. Všechny příklady a cvičení v této příručce používají bash.

csh nebo C shell: syntaxe tohoto shellu připomíná jazyk C. Mají jej v oblibě někteří pro-gramátoři.
tcsh nebo Turbo C shell: nadmnožina standardního C shellu, uživatelsky příjemnější a rychlejší.
ksh nebo Korn shell: někdy bývá oceňován lidmi s unixovými kořeny. Nadmnožina Bour-ne shellu, ve standardní konfiguraci noční můra začínajících uživatelů.

V souboru /etc/shells naleznete seznam shellů, které daný linuxový systém zná:

```
mia:~> cat /etc/shells /bin/bash /bin/sh /bin/tcsh /bin/csh
```

Výchozí shell uživatele je definován v souboru /etc/passwd, například takto vypadá záznam pro uživatele *mia*:

```
mia:L2NOfqdlPrHwE:504:504:Mia Maya:/home/mia:/bin/bash
```

Přepnutí z jednoho shellu do druhého provedete jednoduše tak, že v aktivním terminálovém okně zadáte název nového shellu. Systém podle hodnoty proměnné PATH nalezne adresář, ve kterém je požadovaný shell uložen, a protože shell sám je spustitelný soubor (program), aktuální shell jej aktivuje a spustí. Obvykle se zobrazí nový prompt, protože každý shell má svůj typický vzhled:

```
mia:~> tcsh [mia@post21 ~]$
```

Výhody Bourne Again Shellu

Bash je GNU shell

Projekt GNU (GNU's Not UNIX) nabízí nástroje pro administraci systému unixového typu, jde o svobodné programy odpovídající unixovým standardům. Bash je kompatibilní s Bourne shellem a převzal užitečné funkce z Korn shellu a C shellu. Snaží se vyhovovat standardu IEEE POSIX P1003.2/ISO 9945.2 Shell and Tools. Oproti Bourne shellu nabízí vylepšení jak pro potřeby programování, tak při interaktivním použití. Mezi tato vylepšení patří například možnost editace příkazového řádku, historie příkazů s neomezenou délkou, řízení úloh, funkce a aliasy, indexovaná pole s neomezenou velikostí a celočíselná aritmetika se zákla-dem 2 až 64. Pomocí bashe lze bez úprav spustit většinu skriptů určených pro Bourne shell. Stejně jako ostatní GNU projekty byl vývoj bashe zahájen s cílem zachovat, chránit a dále šířit svobodu užívání, studování, kopírování, modifikování a redistribuce programů. Je obecně známo, že takové podmínky podněcují kreativitu. Ukazuje se to i na bashi, který nabízí celou řadu dalších funkcí navíc oproti ostatním shellům.

Funkce dostupné jen v bashi Spouštění

Kromě jednoznakových řádkových voleb pro spouštění shellu, které je možné obecně nastavovat pomocí vestavěného příkazu set, existuje i několik víceznakových voleb. V této a následujících kapitolách se zmíníme o několika často používaných volbách, jejich úplný seznam naleznete na informačních stránkách bashe, Bash features -> Invoking Bash.

Inicializační skripty shellu

Inicializační skripty jsou skripty, které bash čte a provádí při svém spouštění. V následujících oddí-lech popisujeme různé způsoby spuštění shellu a z toho vyplývající použité inicializační soubory.

Spuštění jako interaktivní přihlašovací shell nebo s volbou -- login

Interaktivní znamená, že můžete zadávat příkazy. Shell tedy není spuštěn proto, že by byl aktivován nějaký skript. Přihlašovací znamená, že jste shell „získali“ po autentizaci do systému, typicky zadáním jména a hesla.

Načítané soubory:

- /etc/profile
- ~/.bash_profile, ~/.bash_login nebo ~/.profile: přečte se první nalezený soubor
- ~/.bash_logout při odhlášení

Pokud konfigurační soubor existuje, ale nelze jej číst, vypíše se chybové hlášení. Pokud soubor neexistuje, hledá bash následující soubor.

Spuštění jako interaktivní nepřihlašovací shell

Nepřihlašovací shell znamená, že se nemusíte systému autentizovat. Nepřihlašovací shell získáte například při otevření terminálového okna.
Načítané soubory:

- ~/.bashrc Na tento soubor se typicky odkazuje ze souboru ~/.bash_profile:

```
if [ -f ~/.bashrc ]; then . ~/.bashrc; fi
```

Více informací o konstrukci if naleznete v kapitole „Podmíněné příkazy“.

Neinteraktivní spuštění

Všechny skripty používají neinteraktivní shell. Jsou naprogramovány tak, aby dělaly určitou věc, a nelze jim říct, aby dělaly něco jiného.
Načítané soubory:

- definováno v BASH_ENV

Při hledání definovaných souborů se nepoužívá proměnná PATH, takže je rozumné soubory vždy zadávat s uvedením plné cesty.

Spuštění příkazem sh

Bash se snaží chovat stejně jako historický program sh, zároveň se ale snaží vyhovovat standar-du POSIX.Načítané soubory:

```
/etc/profile
~/.profile
```

Při interaktivním spuštění může na další inicializační informace odkazovat proměnná ENV.

Režim POSIX

Tato volba se zapíná buď vestavěným příkazem set:

```
set -o posix
```

anebo voláním programu bash s volbou -- posix. Bash se bude co nejvíce snažit vyhovět posix-ovému standardu. Stejný efekt má nastavení proměnné POSIXLY_CORRECT.Načítané soubory:

- definované proměnnou ENV

Vzdálené spuštění

Soubory načítané při spuštění příkazem rshd:

- ~/.bashrc

Vyhýbejte se použití r-nástrojů Nezapomínejte na nebezpečí spojená s používáním příkazů jako rlogin, telnet, rsh a rcp. Tyto příkazy jsou ze své podstaty nebezpečné, protože posílají citlivá data po síti nešifro-vaně. Pokud potřebujete nástroje pro vzdálené spouštění příkazů, přenos souborů a podobně, použijte implementaci Secure SHell, známou obecně jako SSH, která je volně k dispozici na *http://www.openssh.org.* Existují i různé klientské programy pro neunixové operační systémy.

Spouštění, je-li UID různé od EUID

V tomto případě se nezpracovávají žádné inicializační soubory.

Interaktivní shelly

Co je to interaktivní shell?

Interaktivní shell obecně čte a zapisuje na uživatelský terminál: vstup a výstup jsou spojeny s ter-minálem. V bashi se interaktivní chování aktivuje v případě, že příkaz bash spustíte bez dalších parametrů, bez volby pro čtení vstupu ze souboru, případně s explicitně nastaveným čtením ze standardního vstupu, což umožní nastavení pozičních parametrů (viz kapitolu „Prostředí bashe“).

Je shell interaktivní?

Můžete to zjistit přečtením hodnoty speciálního parametru -, je-li shell interaktivní, obsahuje znak „i“:

```
eddy:~> echo $-himBH
```

V neinteraktivním shellu není nastavena proměnná PS1, prompt.

Chování interaktivního shellu

Rozdíly platné pro interaktivní režim:

Bash načítá inicializační soubory.
Standardně je zapnuto řízení úloh.
Je nastaven prompt a proměnná PS2 pro víceřádkové příkazy, typicky má hodnotu „>“. Tento prompt uvidíte také v případě, že zadáte neúplný příkaz – například zapomenete-li uzavírací uvozovku, neuzavřete příkazovou strukturu a podobně.
Příkazy se načítají z příkazového řádku prostřednictvím readline.
Bash interpretuje volbu ignoreeof namísto okamžitého ukončení po přijetí EOF (konec souboru).
Standardně je zapnuta historie příkazů a expanze historie. Při ukončení shellu se historie zapíše do souboru definovaného v HISTFILE, standardně se používá soubor ~/.bash_history.
Je zapnuta expanze aliasů.
Při nepřítomnosti trapů se ignoruje signál SIGTERM.
Při nepřítomnosti trapů se zachycuje a obsluhuje signál SIGINT. Proto například stisk Ctrl+C nezpůsobí ukončení interaktivního shellu.
Volbou huponexit je standardně nastaveno zaslání signálu SIGHUP všem spuštěným úlo-hám při ukončení shellu.
Příkazy se provádějí ihned po načtení.
Bash pravidelně kontroluje došlou poštu.
Bash lze nastavit tak, aby se při odkazu na dereferencovanou proměnnou ukončil. V interaktivním režimu je toto chování vypnuto.
Zaznamená-li vestavěný příkaz chybu přesměrování, nedojde k ukončení shellu.
Vrátí-li v režimu POSTFIX speciální vestavěný příkaz chybu, nedojde k ukončení shellu. Speciální vestavěné příkazy jsou uvedeny v kapitole „Ladění skriptů“.
Selhání příkazu exec nezpůsobí ukončení shellu.
Syntaktické chyby parseru nezpůsobí ukončení shellu.
Je zapnuta jednoduchá kontrola parametrů vestavěného příkazu cd.
Je podporování automatické ukončení po uplynutí doby nastavené v proměnné TMOUT.

Další informace viz:

Kapitolu „Proměnné“.
Kapitolu „Aliasy“.
Více informací o signálech obsahuje kapitola „Zpracování signálů“.

V kapitole „Expanze shellu“ jsou popisovány různé expanze prováděné po zadání příkazu.

Podmínky

Podmíněné výrazy používá složený příkaz [[a vestavěné příkazy test a [.
Výraz může být unární nebo binární. Unární výrazy se často používají ke zjištění stavu souboru.
Operace se provádí nad jediným objektem, například souborem.
Dále jsou k dispozici operátory pro porovnávání řetězců a číselných hodnot. Tyto operátory jsou

binární, vyžadují dva objekty, nad nimiž se operace provede. Pokud jako parametr FILE někte-rého ze souborových operátorů zadáte /dev/fd/N, bude se testovat souborový deskriptor
N. Pokud zadáte dev/stdin, /dev/stdout nebo /dev/stderr, bude se testovat souborovýdeskriptor 0, 1, respektive 2.
O podmínkách hovoříme v kapitole „Podmíněné příkazy“.Podrobnější informace o souborových deskriptorech naleznete v kapitole „Přesměrována deskriptory souborů“.

Aritmetika v shellu

Shell umožňuje vyhodnocovat aritmetické výrazy, ať už prostřednictvím některé z expanzí shellu

nebo vestavěným příkazem let.Výpočet probíhá celočíselně s pevně danou velikostí čísla, nekontroluje se přetečení – výjimkouje dělení nulou, které je zachyceno a hlášeno jako chyba. Operátory a jejich priorita a asociativi-ta jsou stejné jako v jazyce C, viz kapitolu „Prostředí bashe“.

Aliasy

Aliasy umožňují nahradit slovo řetězcem, je-li slovo použito jako první slovo jednoduchého pří
kazu. Seznam shellem udržovaných aliasů je možno nastavovat a rušit příkazy alias a unalias. Před provedením jakéhokoliv příkazu na řádku načítá bash vždy minimálně alespoň jeden celý řádek vstupu. Aliasy se expandují při čtení příkazu, nikoliv při jeho spuštění. Pokud se tedy na jednom řádku vyskytuje definice aliasu a další příkaz, definice se neuplatní, dokud nebude načten další řádek vstupu. Příkaz bezprostředně následující za aliasem na stejném řádku tak nebude nastavením aliasu ovlivněn.

Aliasy se expandují při čtení definice funkce, nikoliv při volání funkce, protože sama definice funkce představuje složený příkaz. Důsledkem je, že aliasy definované ve funkci nebudou k dispozici, dokud nebude funkce volána.

Podrobněji o aliasech hovoříme v kapitole „Aliasy“.

Pole

Bash podporuje jednorozměrná pole. Jako pole je možno použít libovolnou proměnnou, k expli-citní deklaraci slouží vestavěný příkaz declare. Velikost pole není omezena, rovněž nejsou žádné požadavky na volbu indexů prvků či jejich spojitost. Pole se indexují od nuly. Viz kapitolu „Více
o proměnných“.

Adresářový zásobník

Adresářový zásobník je seznam naposledy navštívených adresářů. Vestavěný příkaz pushd přidá-vá zadaný adresář na zásobník a změní na něj aktuální adresář, příkaz popd odstraňuje adresář ze zásobníku a změní aktuální adresář na odstraněný.

Obsah zásobníku je možno zobrazit příkazem dirs nebo jako obsah proměnné DIRSTACK. Více informací o chování tohoto mechanismu naleznete na informačních stránkách bashe.

Prompt

Bash umožňuje libovolně měnit vzhled promptu. Viz část *Controlling the Prompt* v informačních stránkách bashe.

Omezený shell

Pokud shell spustíte příkazem rbash nebo s volbami --restricted nebo -r, aktivují se následují-cí omezení:

Vypne se vestavěný příkaz cd.
Nebude možné měnit obsah proměnných SHELL, PATH, ENV nebo BASH_ENV.
Názvy příkazů nebudou moci obsahovat lomítka.
S vestavěným příkazem . (a source) nebude možno použít názvy souborů obsahující lomítka.
Vestavěný příkaz hash neumožní použít lomítka s volbou -p.
Při spuštění se vypíná se import funkcí.
Při spuštění se ignoruje nastavení SHELLOPTS.
Není možné provádět přesměrování prostřednictvím operátorů >, >|, ><, >&, &> a >>.
Je vypnutý vestavěný příkaz exec.
Vestavěný příkaz enable má vypnuty volby -f a -d.

U vestavěného příkazu command nelze použít výchozí nastavení PATH.
Omezený režim nelze vypnout.

Pokud je v omezeném shellu spuštěn shellový skript, proběhne v subshellu, který nebude omezen. Více informací viz:
kapitolu „Proměnné“,
kapitolu „Další volby bashe“,
Info Bash -> Basic Shell Features -> Redirections,
kapitolu „Přesměrování a deskriptory souborů“ – pokročilé přesměrování.

Spouštění příkazů

Obecné

U spouštěných programů bash zjišťuje jejich typ. Řada systémových příkazů je realizována nor-málními programy, které jsou v systému v binární podobě uloženy. Spouštíte-li takovýto program, vznikne nový proces, protože bash vytvoří svou vlastní přesnou kopii. Tento synovský proces má nastaveno stejné prostředí jako rodič, liší se pouze identifikačním číslem procesu. Tento postup se označuje jako *forking*.

Jakmile se bash „forkne“, adresní prostor synovského procesu bude přepsán daty nového proce
su. To zajišťuje systémové volání *exec*. Mechanismus *fork-a-exec* tedy nahradí starý příkaz novým, přičemž prostředí, v němž nový pro-gram běží, zůstává identické včetně konfigurace vstupních a výstupních zařízení, proměnných pro-středí a priority. Tento mechanismus se používá při vytváření všech unixových procesů, a platí tedy i v operačním systému Linux. Dokonce i první proces, init, s ID procesu 1, je forkován při bootovacím procesu v proceduře označované jako *bootstrapping.*

Vestavěné příkazy shellu

Vestavěné příkazy jsou přímo součástí shellu. Uvedete-li jako první slovo jednoduchého příkazu název vestavěného příkazu, provede shell tento příkaz přímo, nedojde k vytvoření nového pro-cesu. Vestavěné příkazy jsou nezbytné k implementaci funkcí, jejichž realizace samostatnými pro-gramy by byla nemožná nebo nepohodlná.

Bash podporuje tři typy vestavěných příkazů:

- Vestavěné příkazy Bourne shellu:

  :, ., break, cd, continue, eval, exec, exit, export, getopts, hash, pwd, readonly, return, set, shift, test, [, times, trap, umask a unset.

- Vestavěné příkazy bashe:

  alias, bind, builtin, command, declare, echo, enable, help, let, local, logout, printf, read, shopt, type, typeset, ulimit a unalias.

- Speciální vestavěné příkazy:

  Je-li bash spuštěn v režimu POSIX, liší se chování speciálních vestavěných příkazů od ostat-ních vestavěných příkazů v následujících bodech:

Názvy speciálních vestavěných příkazů se detekují dříve než názvy funkcí shellu.
Skončí-li speciální vestavěný příkaz chybou, neinteraktivní shell bude ukončen. Příkazy přiřazení předcházející příkazu zůstanou po skončení příkazu v platnosti.

Speciální posixové vestavěné příkazy jsou :, ., break, continue, eval, exec, exit, export, readonly, return, set, shift, trap a unset.

O většině uvedených vestavěných příkazů budeme hovořit v dalších kapitolách. Další informace o těch příkazech, o nichž se nebudeme zmiňovat, naleznete na informačních stránkách.

Spouštění programů ze skriptu

Jestliže spustíte skript, vytvoří bash voláním *fork* nový proces bashe. Tento subshell čte řádky ze skriptu jeden po druhém. Po načtení řádku následně přečte, interpretuje a provede příkazy na tomto řádku stejně, jako kdyby byly zadávány přímo z klávesnice.

Zatímco subshell zpracovává jednotlivé řádky skriptu, jeho rodičovský shell čeká, než synovský proces skončí. Jakmile je skript zpracován celý, subshell se ukončí. Tím dojde k probuzení rodičovského shellu, který zobrazí prompt.

Stavební bloky

Stavební bloky shellu Syntaxe shellu

Jestliže vstup není komentářem (komentář začíná znakem # a pokračuje až do konce řádku), shell jej přečte a rozdělí na slova a operátory, přičemž ke zjištění významu jednotlivých znaků na vstu-pu používá přepisovacích pravidel. Následně jsou slova a operátory přeloženy na příkazy a další konstrukce, které vracejí návratový kód, jejž je možno následně kontrolovat či zpracovat. Výše popsaný mechanismus fork-a-exec vstupuje do hry až ve chvíli, kdy shell provedl následující ana-lýzu vstupu:

Shell čte vstup ze souboru, z řetězce nebo z uživatelského terminálu.
Ve shodě s přepisovacími pravidly (viz kapitolu „Prostředí bashe“) je vstup rozdělen na slova a operátory. Tyto tokeny jsou od sebe odděleny pomocí *metaznaků*. Dojde k expan-zi aliasů.
Shell *parsuje* (analyzuje a substituuje) tokeny na jednoduché a složené příkazy.
Bash provede různé expanze shellu, kdy dojde k rozdělení expandovaných tokenů na seznamy souborů, příkazů a parametrů.
V případě potřeby se provede přesměrování, operátory přesměrování a jejich operandy jsou odstraněny ze seznamu parametrů.
Provedou se příkazy.
Shell případně čeká na dokončení příkazu a převezme jeho návratový kód.

Příkazy shellu

Jednoduchý příkaz shellu, například touch soubor1 soubor2 soubor3 se skládá ze samotného příkazu, za nímž následují parametry oddělené mezerami.
Složitější příkazy jsou složeny z jednoduchých příkazů vzájemně spojených různými způsoby: Pomocí roury, kdy se výstup jednoho příkazu stává vstupem jiného, pomocí smyček nebo podmíněných konstrukcí nebo jinými způsoby. Několik příkladů:

ls | more gunzip file.tar.gz | tar xvf -

Funkce shellu

Funkce shellu představuje způsob, jak seskupit více příkazů tak, aby je bylo možné později všech
ny spustit voláním jednoho názvu skupiny. Příkazy se provádějí úplně stejně jako jindy. Pokud
jako název příkazu zadáte název funkce shellu, provede se seznam příkazů asociovaných
s názvem příslušné funkce.

Funkce shellu se provádějí v kontextu aktuálního shellu, k jejich interpretaci se nevytváří novýproces.O funkcích budeme hovořit v kapitole „Funkce“.

Parametry shellu

Parametr je entita, která obsahuje hodnotu. Může jít o název, číslo nebo nějakou speciální hod-notu. Pro účely shellu chápeme proměnnou jako parametr, který obsahuje název. Každá pro-měnná má hodnotu a žádný nebo více atributů. Proměnné se vytvářejí vestavěným příkazem dec-lare.

Pokud není zadána hodnota, obsahuje proměnná prázdný řetězec. Proměnné lze zrušit pouze

voláním vestavěného příkazu unset.Přiřazování proměnných je popsáno v kapitole „Proměnné“, složitější operace s proměnnými jsoupopsány v kapitole „Více o proměnných“.

Expanze shellu

K expanzi dochází poté, co je příkazový řádek rozdělen na tokeny. Následně se provádějí tyto expanze:

expanze složených závorek,
expanze tildy,
expanze parametrů a proměnných,
substituce příkazů,
aritmetické expanze,
dělení slov,
expanze názvů souborů.

O jednotlivých expanzích budeme podrobněji hovořit v kapitole „Expanze shellu“.

Přesměrování

Při spouštění příkazů je možné prostřednictvím speciální notace interpretované shellem přesměrovat jejich vstup a výstup.
Přesměrování lze také použít k otevření a zavření souborů v prováděcím prostředí aktuálního shellu.

Provádění příkazů

Při provádění příkazů se nejprve pro pozdější použití uloží slova, která parser označil jako přiřazení proměnných (tedy

předcházející názvu příkazu) a nastavení přesměrování. Slova, která nejsou přiřazeními a přesměrováními, se expandují. První slovo vzniklé po expanzi je chápáno jako název příkazu, ostatní pak jako jeho parametry. Poté se provede přesměrování a expandují se řetězce přiřazené proměnným. Není-li výsledkem název žádného příkazu, proměnné ovlivní prostředí aktuálního shellu.

Důležitým úkolem shellu je nalezení příkazu. Bash to provádí takto:

Ověří, zda název příkazu obsahuje lomítka. Pokud ne, zkontroluje nejprve seznam funkcí, zda neobsahuje požadovaný název.
Není-li příkaz funkcí, hledá se v seznamu vestavěných příkazů.
Není-li příkaz ani funkcí ani vestavěným příkazem, hledá se v adresářích definovaných v proměnné PATH. K uložení názvů příkazů i s celými cestami používá bash *hashovací tabulku* (datovou strukturu uloženou v paměti), nedochází tak k opakovanému náročné-mu prohledávání všech adresářů v cestě.
Jestliže se nepodaří příkaz nalézt, vypíše bash chybové hlášení a vrátí návratový kód 127.
Pokud bylo hledání úspěšné nebo pokud název příkazu obsahoval lomítka, spustí shell příkaz v samostatném prováděcím prostředí.
Pokud se spuštění nepodaří, protože soubor není spustitelný a není to adresář, bude považován za shellový skript.
Pokud příkaz nebyl spuštěn asynchronně, shell čeká na jeho skončení a převezme jeho návratový kód.

Shellové skripty

Pokud při spuštění bashe (bez voleb -c nebo -s) uvedete jako první parametr název souboru obsa-hujícího příkazy shellu, dojde ke spuštění neinteraktivního shellu. Tento shell hledá požadovaný skript nejprve v aktuálním adresáři, a pokud tam není, prohledává adresáře uvedené v proměnné PATH.

Tvorba dobrých skriptů

Vlastnosti dobrého skriptu

V této příručce budeme hovořit zejména o posledním ze stavebních bloků, o skriptech. Než budeme pokračovat, uveďme si několik obecných doporučení:

Skript má běžet bez chyb.
Skript má dělat to, k čemu je určen.
Logika programu má být jasně definovaná a zřejmá.
Skript nemá dělat nic zbytečného.
Skript má být použitelný univerzálně.

Struktura

Struktura skriptu je velmi flexibilní. I když vám bash nechává velkou míru svobody, musíte zajis-tit logickou správnost, řízení běhu a efektivitu tak, aby uživatelé mohli skripty používat snadno a korektně.

Když začínáte pracovat nad novým skriptem, odpovězte si na následující otázky:

Budu potřebovat nějaké informace od uživatele nebo z uživatelského prostředí?
Jak tyto informace uložím?
Bude nutné vytvářet nějaké soubory? Kde a s jakými právy a vlastníky?
Jaké příkazu budu používat? Bude-li se skript používat na různých systémech, obsahují tyto systémy potřebné příkazy v potřebné verzi?
Bude nutné uživateli něco sdělit? Kdy a proč?

Terminologie

Následující tabulka představuje přehled programátorských termínů, kterým byste měli rozumět:

Termín	Co to znamená?
řízení příkazu	Testování návratového kódu příkazu za účelem zjištění, zda má být provedena nějaká část pro gramu.
podmíněná větev	Logické místo v programu, kde podmínka rozhoduje, co se stane dál.
prováděcí logika	Celkový návrh programu. Určuje logickou posloupnost kroků k úspěšnému dosažení správného výsledku.
smyčka	Část programu, která se provede nula nebo vícekrát.

uživatelský vstup	Informace získané z vnějšího zdroje v době běhu programu, je možno je uložit a podle potřeby použít.

Přehled programátorských termínů

K pořadí a logice

Kvůli urychlení práce při vývoji skriptu je nutné dopředu promyslet logické uspořádání programu.
Jde vždy o první krok při vývoji.Lze použít různé metody, nejobvyklejší je použití seznamů. Vyjmenováním úkonů prováděnýchv programu můžete jednotlivé fáze snadno popsat. Na jednotlivé kroky se pak můžete odkazovatjejich číslem v seznamu. Jednotlivé prováděné úkony sepisujete v běžné „lidské terminologii“, což vám usnadní tvorbu sro

zumitelného programu. Později nahradíte lidské výrazivo příkazy a konstrukcemi shellu.
Následující příklad ukazuje návrh logického uspořádání. Popisuje systém rotace logů. Příklad demonstruje použití smyčky, která je prováděna přes jednotlivé soubory, které chcete rotovat.

1. Chceme rotovat logy?a) Pokud ano:
    i. Zadej název adresáře s rotovanými logy.
    ii. Zadej název souboru s logem.
    iii. Zadej počet dní, po které má být log uchováván.
    iv. Proveď trvalé nastavení v crontabu uživatele.
   b) Pokud ne, běž na krok 3.

Chceme rotovat další skupinu logů?
a) Pokud ano: opakuj krok 1.
b) Pokud ne: jdi na krok 3.

Konec.

Uživatel musí programu poskytnout nějaké informace. Tyto údaje je nutné nějak získat a uložit. Uživatele bychom měli upozornit, že dojde ke změně jeho crontabu.

Příklad shellového skriptu: mysystem.sh

Následující skript mysystem.sh pomocí několika známých příkazů (date, w, uname, uptime) vypíše základní informace o vás a vašem počítači.

```
tom:~> cat -n mysystem.sh
     1 #!/bin/bash
     2 clear
     3 echo "Následující údaje vám přináší mysystem.sh. Právě začínáme."
     4
     5 echo "Ahoj, $USER"
     6 echo
     7
     8 echo "Dnes je `date`, je to `date +"%V"`. týden."
     9 echo
    10
    11 echo "Právě jsou přihlášeni následující uživatelé:"
    12 w | cut -d " " -f 1 - | grep -vUSER | sort -u
    13 echo
    14
    15 echo "Systém `uname -s` běží na platformě `uname -m`."
    16 echo
    17
    18 echo "Doba běhu systému:"
    19 uptime
    20 echo
    21
    22 echo "A to je vše!"
```

Každý skript vždy začíná stejnou dvojicí znaků, „#!“. Za nimi následuje název shellu, kterým mají být provedeny dále zadané příkazy. Skript nejprve na druhém řádku vymaže obrazovku. Třetí řádek vypíše zprávu o tom, co se bude dít. Pátý řádek uživatele

pozdraví. Řádky 6, 9, 13, 16 a 20 mají za úkol pouze správně formátovat vypisovaný text. Osmý řádek vypíše dnešní datum a číslo týdne. Řádek 11 je opět informativní zpráva, stejně jako řádky 3, 18 a 22. Řádek 12 formátuje výstup příkazu w, řádek 15 zobrazí název operačního systému a procesor. Řádek 19 vypíše infor-mace o době běhu a zatížení systému.

Příkazy echo a printf jsou vestavěné příkazy bashe. První z nich skončí vždy návratovým kódem 0 a jednoduše na standardní výstup vypíše všechny své parametry až po konec řádku. Druhý z nich umožňuje definovat formátovací řetězec a v případě chyby vrací nenulový návratový kód.

Stejný skript s použitím příkazu printf bude vypadat takto:

```
#!/bin/bash clear printf "Následující údaje vám přináší mysystem.sh. Právě začínáme."

printf "Ahoj, $USER.\n\n"

printf "Dnes je `date`, je to `date +"%V"`. týden.\n\n"

printf "Právě jsou přihlášeni následující uživatelé:\n" w | cut -d " " -f 1 - | grep -v USER | sort -u printf "\n"

printf "Systém `uname -s` běží na platformě `uname -m`.\n\n"
printf "Doba běhu systému:\n"
uptime
printf "\n"


printf "A to je vše!\n"
```

V kapitole „Tvorba interaktivních skriptů“ budeme podrobněji hovořit o tom, jaké informativní údaje má skript poskytovat tak, aby byl uživatelsky příjemný.

Standardní umístění bashe

Z příkladu vyplývá, že program bash je standardně umístěn v adresáři /bin.

Není-li k dispozici stdout

Pokud skript spouštíte z cronu, uvádějte plné cesty k příkazům a přesměrujte výstup a chy bový výstup. Shell běží v neinteraktivním režimu, takže v případě jakékoliv chyby dojde k předčasnému ukončení skriptu.

Podrobnější informace o konstrukcích použitých ve výše uvedených příkladech naleznete v dalších kapitolách.

Příklad inicializačního skriptu

Inicializační skript spouští na unixových a linuxových strojích systémové služby. Typickými příkla-dy takových služeb jsou logovací démon, démon pro správu napájení nebo démony pošty či služ-by DNS. Inicializační skripty služeb, označované též jako spouštěcí skripty, jsou v systému ulože-ny na specifickém místě, typicky v adresáři /etc/rc.d/init.d nebo /etc/init.d. Proces init, první proces systému, přečte své konfigurační soubory a rozhodne, které služby na konkrétní úrov-ni běhu spustit a zastavit. Úroveň běhu je vlastně konfigurace procesů; každý systém má například definovánu jednouživatelskou úroveň běhu pro potřeby administrativních úkonů, při nichž je nutné, aby v systému běželo co nejméně věcí – například při obnově kritických souborových systé-mů ze zálohy. Další běžně definované úrovně jsou úrovně pro restart a vypnutí systému.

Úkony prováděné při spouštění a zastavování služby jsou definovány právě v odpovídajícím ini-cializačním skriptu. Jedním z úkolů správce systému je nastavit init tak, aby byly služby spouště-ny a zastavovány ve správném pořadí. V těchto případech musíte dobře rozumět proceduře spouš-tění a zastavování systému. Než tedy začnete vytvářet vlastní inicializační skripty, doporučujeme vám přečíst si manuálové stránky init a inittab.

Takto vypadá velmi jednoduchý příklad, který při spuštění a zastavení počítače přehraje zvukový soubor:

```
#!/bin/bash

# Skript do /etc/rc.d/init.d # linkován jako rc3.d/S99audio-greeting a rc0.d/K01audio-greeting

case "$1" in
'start')
```

```
    cat /usr/share/audio/at_your_service.au > /dev/audio
    ;; 'stop')
    cat /usr/share/audio/oh_no_not_again.au > /dev/audio
    ;; esac exit 0
```

Příkaz case, který se v tomto typu skriptů velmi často používá, popisujeme v kapitole „Použití pří-kazů if a exit“.

Shrnutí

Bash je GNU shell kompatibilní s Bourne shellem, který obsahuje celou řadu užitečných funkcí z jiných shellů. Když je shell spuštěn, čte konfigurační soubory. Mezi nejdůležitější patří:

/etc/profile,
~/.bash_profile,
~/.bashrc.

Chování bashe se může lišit podle toho, zda je spuštěn v interaktivním režimu, režimu POSIX
a v omezeném režimu.Příkazy shellu můžeme rozdělit do tří kategorií: funkce shellu, vestavěné příkazy shellu a příkazyexistující v adresářích souborového systému. Bash podporuje různé vestavěné příkazy, kterénejsou součástí Bourne shellu.

Shellové skripty se skládají z těchto typů příkazů uspořádaných tak, jak požaduje syntaxe shellu. Skripty se čtou a provádějí řádek po řádku a měly by mít logickou strukturu.

Cvičení

Několik cvičení, která vás zahřejí před následující kapitolou:

Kde se ve vašem systému nachází program bash? Pomocí volby -- version zjistěte, jakou verzi shellu používáte.
Které konfigurační soubory shellu se načítají, když se k systému přihlásíte prostřednictvím grafického rozhraní a poté otevřete terminálové okno?
Jsou následující shelly interaktivní? Jsou přihlašovací?

Shell otevřený tím, že na pozadí pracovní plochy klepnete pravým tlačítkem myši a v nabídce vyberete příkaz „Otevřít terminál“ či podobný.
Shell, který získáte příkazem ssh *localhost*.
Shell, který získáte přihlášením na konzolu v textovém režimu.
Shell, který získáte příkazem xterm &.
Shell, otevřený skriptem mysystem.sh.
Shell, který získáte na vzdáleném systému, k němuž nemáte jméno a heslo, jelikož pou-žíváte SSH a klíče.

Dokážete vysvětlit, proč bash neskončí, zmáčknete-li na příkazovém řádku Ctrl+C? Zob-razte obsah zásobníku adresářů.
Pokud už takové nastavení nemáte, nastavte prompt tak, aby vám zobrazoval aktuální adresář. Do souboru ~/.bashrc můžete přidat například takovýto řádek:

```
    export PS1="\u@\h \w> "
```

8. Vypište hashované příkazy aktuální relace shellu. Kolik procesů momentálně ve vašem systému běží? Použijte příkazy ps a wc, první řádek výpisu ps není proces! Jak zobrazíte název vašeho počítače? Jen název, nic jiného!

Tvorba a ladění skriptů

Po přečtení této kapitoly budete umět:

Napsat jednoduchý skript
Určit typ shellu, v němž má být skript spuštěn
Uvádět ve skriptu komentáře
Změnit práva skriptu

Spustit a odladit skript

Vytvoření a spuštění skriptu

Psaní a pojmenování

Shellový skript je sekvence příkazů, kterou opakovaně používáte. Sekvence se typicky spustí zadá-ním názvu skriptu na příkazovém řádku. Alternativně lze skripty používat k automatickému pro-vádění úkonů prostřednictvím cronu. Další použití skriptů je ve spouštěcí a zastavovací sekvenci systému, kde se prostřednictvím inicializačních skriptů spouštějí démoni a služby.

Chcete-li vytvořit shellový skript, otevřete ve svém oblíbeném editoru nový prázdný soubor. Lze použít libovolný textový editor – vim, emacs, gedit, dtpad a další. Doporučujeme vám ovšem používat mocnější editory typu vim nebo emacs, protože je lze nastavit tak, aby rozeznaly syn-taxi shellového skriptu, což je velmi pohodlná funkce, která vám zabrání v běžných chybách, jako jsou zapomenuté závorky či středníky.

V novém souboru pište příkazy na jednotlivé řádky tak, jako byste je přímo zadávali v příkazo-vém řádku. Jak už jsme uvedli (viz kapitolu „Spouštění příkazů“), příkazy mohou být funkce, vestavěné příkazy, externí příkazy a jiné skripty.

Skript pojmenujte vhodným názvem, který bude vyjadřovat, co skript dělá. Ověřte si, že zvolený název nekoliduje s již existujícími příkazy. Aby se předešlo kolizím, často názvy skriptů končí znaky .sh; i tak se ale může stát, že v systému existuje skript se stejným názvem, jaký jste si zvo-lili. Informace o programech a souborech můžete ověřit pomocí příkazů which, whereis a dal-ších:

which -a název_skriptu whereis název_skriptu
locate název_skriptu

script1.sh

V tomto příkladu používáme vestavěný příkaz echo, jímž uživatele nejprve informujeme o tom, co vypíšeme, a následně zavoláme příkaz, který příslušný výpis provede. Vřele doporučujeme informovat uživatele o tom, co skript dělá, aby pak nebyli zbytečně nervózní z pocitu, že *skript nedělá nic*. Na téma informování uživatelů se budeme bavit více v kapitole „Tvorba interaktivních skriptů“.

Zkuste si tento skript sami napsat. Možná bude rozumné, když si pro ukládání vlastních skriptů vytvoříte adresář ~/scripts. Přidejte tento adresář do proměnné PATH:

export PATH="$PATH:~/scripts"

Pokud s bashem začínáte, rozhodně vám doporučujeme použít editor, který různé konstrukce shellu zobrazuje různými barvami. Zvýrazňování syntaxe podporují vim, gvim, (x)emacs, kwri-te a celá řada dalších – ověřte si to v dokumentaci svého oblíbeného editoru.

> Různé prompty
>
> Prompt, zobrazovaný v různých příkladech v této příručce, může vypadat různě podle momentálního rozpoložení autora. Je to mnohem podobnější realitě než klasický učebni-cový prompt *$*. Jediná dodržovaná konvence je, že prompt *roota* končí vždy znakem #.

Spuštění skriptu

Aby bylo možné skript spustit, musí mít vhodně nastavena práva. Při nastavování práv si vždy ověřte, zda jste skutečně dostali to, co jste zamýšleli. Pak už můžete skript spustit jako kterýkoliv jiný příkaz:

[jura@jv scripts]$ chmod u+x script1.sh

[jura@jv scripts]$ ls -l script1.sh -rwxrw-r--1 jura jura 331 čec 23 12:02 script1.sh

[jura@jv scripts]$ script1.sh

Nyní skript začíná. Ahoj, jura!

Nyní zjistím seznam přihlášených uživatelů:

13:22:20 up 13 days, 23:23, 3 users, load average: 0,16, 0,10, 0,05 USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT jura :0 -09:08 ? xdm? 1:13m 0.52s /usr/bin/gnome-session jura pts/2 :0.0 10:38 2:35m 0.03s 0.03s bash jura pts/3 :0.0 11:55 0.00s 0.05s 0.00s /bin/bash ./script1.sh

Nyní nastavuji dvě proměnné. Toto je řetězec: black A toto je číslo: 9

A te už ti vrátím prompt

[jura@jv scripts]$ echo $COLOUR

[jura@jv scripts]$ echo $VALUE

[jura@jv scripts]$

Jde o nejběžnější způsob spuštění skriptu. Doporučujeme skripty jako ten náš spouštět v samo-statném subshellu. Proměnné, funkce a aliasy vytvořené v tomto subshellu tak budou platit pouze v něm. Jakmile skript skončí a řízení se vrátí rodičovskému shellu, všechno bude vyčištěno a skrip-tem provedené změny stavu shellu budou zapomenuty.

Pokud jste adresář scripts nepřidali do proměnné PATH a cesta neobsahuje ani. (aktuální adre-sář), můžete skript spustit takto:

./název_skriptu.sh

Skript můžete také explicitně spustit uvedením názvu shellu, obecně se tento způsob ale používá jen v případech, kdy tím sledujete nějaký speciální záměr – například testujete, zda skript fungu-je i v jiném shellu, nebo pořizujete ladicí výpis:

rbash název_skriptu.sh sh název_skriptu.sh bash -x
název_skriptu.sh
Zadaný shell bude spuštěn jako subshell aktuálního shellu a provede skript. Je to vhodné zejmé-na v případech, kdy potřebujete skript spustit se specifickými volbami nebo specifickým způso-bem, který není ve skriptu definován.

Pokud nechcete spouštět nový shell a chcete skript provést v aktuálním shellu, použijte příkaz source:

source název_skriptu.sh

> source = .Vestavěný příkaz source bashe je ekvivalentní s příkazem . Bourne shellu a užití příkazuje podobné: . název_skriptu.sh.

V takovém případě nemusí mít skript nastaveno spouštěcí právo. Příkazy se provádějí v kontextu aktuálního shellu, veškeré změny provedené v prostředí tak budou viditelné i po skončení skriptu:

[jura@jv scripts]$ source script1.sh -- výstup vynechán --

[jura@jv scripts]$ echo $VALUE 9 [jura@jv scripts]$

Základní informace o skriptech

Který shell skript provede?

Jakmile spouštíte skript v subshellu, měli byste definovat, který shell má skript provést. Shell, pro nějž jste skript napsali, nemusí být výchozím shellem na cílovém systému, takže některé příkazy nemusí v tomto jiném shellu proběhnout správně.

První řádek skriptu definuje, v jakém shellu se má skript provést. První dva znaky tohoto řádku jsou vždy #!, za nimi následuje cesta k shellu, kterým se mají následující příkazy interpretovat. Prázdné řádky se rovněž počítají, skript tedy nemůže začínat prázdným řádkem.

V rámci této příručky budou všechny skripty začínat řádkem:

#!/bin/bash

Jak už bylo řečeno, vyplývá z toho, že spustitelný soubor bashe se nachází v adresáři /bin.

Komentáře

Neměli byste zapomínat, že nemusíte být jedinou osobou, která bude váš skript číst. Řada uživatelů a administrátorů používá skripty, které napsal někdo jiný. Komentáře jim usnadňují pochopit, jak skript funguje.

Komentáře zároveň usnadňují život i autorovi skriptu. Řekněme, že než jste nějaký příkaz použí-vaný ve skriptu přiměli udělat přesně to, co jste potřebovali, stálo vás to hodně studia manuálo-vých stránek a různých příkladů. Až budete za pár týdnů či měsíců potřebovat skript upravit, nebudete si pamatovat, jak to vlastně funguje! Proto je rozumné v komentáři popsat, co jste vlast-ně udělali, jak jste to udělali a proč jste to udělali.
Vezměte příklad script1.sh, zkopírujte jej do souboru commented-script1.sh a doplňte komentáře tak, aby bylo zřejmé, co skript dělá.
Jakmile shell narazí na znak #, celý zbytek řádku se ignoruje a uvidíte jej pouze při zobrazení souboru se skriptem:

#!/bin/bash # Skript smaže obrazovku, pozdraví a vypíše informace o přihlášených # uživatelích. Na konci nastaví a vypíše dvě proměnné

clear # smazání obrazovky

echo “Nyní skript začíná.”

echo “Ahoj, $USER!” # symbol $ vrací obsah proměnné

echo

echo “Nyn zjist seznam přihlš畴n⌠ch uživatelů:” echo w # ukž畴, kdo je přihlš畴n echo # a co pr疱ě děl

echo “Nyn nastavuji dvě proměnn.” COLOUR=”black” # nastaven lok疝n proměnn VALUE=”9” # nastaven lok疝n proměnn echo “Toto je řetězec: $COLOUR” # v⌠pis obsahu proměnn echo “A toto je č lo: $VALUE” # v⌠pis obsahu proměnn echo

echo “A te už ti vr疸 prompt”

V rozumně napsaném skriptu bývá na začátku uvedeno, co skript dělá. Pro snazší pochopení čin-nosti se následně komentuje chování každého většího logického celku příkazů. Například inicia-lizační skripty v adresáři init.d bývají komentovány velmi podrobně, protože je může číst a modifikovat každý správce systému.

Ladění skriptů

Ladění celého skriptu

Pokud skript nefunguje podle očekávání, je nutné zjistit, proč tomu tak je. Bash nabízí rozsáhlé možnosti ladění skriptů. Nejčastější variantou je spuštění subshellu s volbou -x, čímž se celý skript provede v režimu ladění. Na standardní výstup se vypisují jednotlivé prováděné příkazy poté, co dojde k jejich expanzi, ale předtím, než se provedou.

Takto bude vypadat skript commented-script1.sh spuštěný v režimu ladění. Opět si můžete všimnout, že komentáře se nevypisují:

```
[jura@jv scripts]$ bash -x commented-script1.sh
+ echo 'Nyní skript začíná.' Nyní skript začíná.
+ echo 'Ahoj, jura!' Ahoj, jura!
+ echo
+ echo 'Nyní zjistím seznam přihlášených uživatelů:'

Nyní zjistím seznam přihlášených uživatelů:
+ echo
+ w

  09:25:20 up 19 days, 19:26, 2 users, load average: 0,33, 0,09, 0,03 USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT martina :0 -07:01 ?
xdm? 1:47m 0.35s /usr/bin/gnome-session jura pts/2 1048656940.ip2lo 09:13 1.00s 0.03s 0.00s bash -x commen-ted-script1.sh

+ echo 'Nyní nastavuji dvě proměnné.' Nyní nastavuji dvě proměnné.
+ COLOUR=black
+ VALUE=9
+ echo 'Toto je řetězec: black' Toto je řetězec: black
+ echo 'A toto je číslo: 9' A toto je číslo: 9
+ echo
+ echo 'A te už ti vrátím prompt' A te už ti vrátím prompt
+ echo
```

> Plnohodnotné ladění
>
> Na adrese *http://bashdb.sourceforge.net/* naleznete projekt plnohodnotného debuggeru, který můžete do bashe integrovat.

Ladění částí skriptu

Pomocí vestavěného příkazu set můžete v normálním režimu spustit ty části skriptu, které se cho-vají správně, a ladicí údaje můžete vypisovat jen v problémových částech. Řekněme, že si nejste jisti, jak se ve skriptu commented-script1.sh chová příkaz w.

Inkriminovaný příkaz můžete obklopit následující dvojící příkazů:

set -x # zapínám ladění w # ukáže, kdo je přihlášen set +x # vypínám ladění

Výstup pak bude vypadat takto:

[jura@jv scripts]$ bash commented-script1.sh
Nyní skript začíná.
Ahoj, jura!

Nyní zjistím seznam přihlášených uživatelů:

+ w
10:00:08 up 19 days, 20:00, 2 users, load average: 0,05, 0,10, 0,12 USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT martina :0 -07:01 ? xdm? 1:51m 0.51s /usr/bin/gnome session Nyní nastavuji dvě proměnné. Toto je řetězec: black A toto je číslo: 9

jura pts/2 1048656940.ip2lo 09:13 0.00s 0.03s 0.00s bash commented script1.sh

+ set +x

A te už ti vr疸 prompt

V jednom skriptu můžete ladicí režim zapnout a vypnout vícekrát. Přehled užitečných voleb bashe uvádí následující tabulka:

Krátký zápis	Dlouhý zápis	Význam
		Vypíná generování názvů souborů pomocí metaznaků.
		Vypisuje řádky skriptu tak, jak byly načteny ze vstupu.
		Vypisuje expandované příkazy před jejich provedením.

Přehled ladicích voleb

Volby se aktivují znakem „-“ a deaktivují znakem „+“, tak se nespleťte!

Následující příklad ukazuje použití uvedených voleb na příkazovém řádku:

[jura@jv scripts]$ set -v

[jura@jv scripts]$ ls ls commented-script1.sh script1.sh

[jura@jv scripts]$ set +v set +v

[jura@jv scripts]$ ls * commented-script1.sh script1.sh

[jura@jv scripts]$ set -f

[jura@jv scripts]$ ls * ls: *: není souborem ani adresářem

[jura@jv scripts]$ touch *

[jura@jv scripts]$ ls
* commented-script1.sh script1.sh

[jura@jv scripts]$ rm *

[jura@jv scripts]$ ls commented-script1.sh script1.sh

Příslušné režimy ladění můžete také zapnout přímo ve skriptu tím, že odpovídající volby zadáte na prvním řádku v deklaraci shellu. Jak je v Unixu běžné, volby je možno kombinovat:

#!/bin/bash -xv
Jakmile naleznete chybnou část skriptu, můžete před každý problémový příkaz přidat příkaz echo, takže přesně uvidíte, kde a co přestane fungovat. Například:

```
echo "debug: spouštím příkaz w"; w
```

V komplikovanějších skriptech můžete na různých místech vypisovat obsahy různých proměnných, takže snáze odhalíte chybné chování:

```
echo "VARNAME má hodnotu $VARNAME"
```

Shrnutí

Shellový skript je opakovaně použitelná skupina příkazů uložená ve spustitelném textovém sou

boru. K vytvoření skriptu je možné použít libovolný textový editor.Skripty začínají znaky #!, za nimiž následuje cesta k shellu, v němž se má skript provést. Kvůli sro-zumitelnosti a budoucím úpravám se do skriptů přidávají komentáře. Vždy je lepší komentovatvíce než méně.

Pomocí voleb shellu je možné skripty ladit. Tyto volby lze aktivovat v konkrétních místech skrip-tu nebo je lze použít globálně na celý skript. Běžnou ladicí technikou je také umisťování příkazůecho na strategická místa skriptu.

Cvičení

Toto cvičení vám pomůže vytvořit první shellový skript.

Ve svém oblíbeném editoru vytvořte skript. Skript vypíše cestu k vašemu domovskému adresá-ři, typ používaného terminálu a seznam služeb, které se spouštějí na úrovni běhu 3. (Tip: Použij-te HOME, TERM a ls /etc/rc3.d/S*)
Přidejte do skriptu komentáře.
Přidejte do skriptu informace o tom, co dělá.
Změňte práva skriptu tak, aby jej bylo možné spustit.
Udělejte ve skriptu chyby a sledujte, co se stane, když zkomolíte příkaz, vynecháte první řádek nebo na něm uvedete něco nesmyslného, když zkomolíte názvy proměnných shellu nebo je zapíšete malými písmeny místo velkých. Ověřte, jak na to budou reagovat ladicí nástroje.

Prostředí bashe

V této kapitole budeme hovořit o různých způsobech, jak ovlivnit prostředí bashe:

Editace inicializačních souborů shellu
Použití proměnných
Použití různých způsobů uvození
Aritmetické výpočty
Přiřazení aliasů
Použití expanze a substituce

Inicializační soubory shellu

Celosystémové konfigurační soubory /etc/profile

Když bash spustíte interaktivně s volbou --login nebo když jej spustíte jako sh, načte iniciali-zační nastavení ze souboru /etc/profile. Tento soubor typicky nastavuje proměnné PATH, USER, MAIL, HOSTNAME a HISTSIZE.

Na některých systémech se v /etc/profile nastavuje hodnota umask, na jiných systémech se z tohoto souboru odkazuje na další konfigurační soubory, jako jsou například:

Soubor /etc/inputrc, systémový konfigurační soubor vstupních zařízení, zde se napří-klad nastavuje chování konzoly.
Adresář /etc/profile.d, obsahuje soubory s konfigurací konkrétních programů platnou pro celý systém.

V souboru /etc/profile se uvádí všechna nastavení, která mají platit společně pro všechny uži-vatele. Může vypadat například takto:

```
# /etc/profile

# Celosystémové nastavení prostředí při přihlašování

PATH=$PATH:/usr/X11R6/bin

# vypínáme coredumpy ulimit -S -c 0 > /dev/null 2>&1

USER="`id -un`" LOGNAME=$USER
MAIL="/var/spool/mail/$USER"

HOSTNAME=`/bin/hostname` HISTSIZE=1000

# chování klávesnice, zvonku a displeje načítáme ze samostatného souboru: if [ -z "$INPUTRC" -a ! -f "$HOME/.inputrc" ]; then
      INPUTRC=/etc/inputrc fi

PS1="\u@\h \W"

export PATH USER LOGNAME MAIL HOSTNAME HISTSIZE INPUTRC PS1

    # Inicializační soubory konkrétních programů (ls, vim, less, ...)
    for i in /etc/profile.d/*.sh ; do if [ -r "$i" ]; then . $i fi done

# Nastavení pro inicializaci programů source /etc/java.conf export
NPX_PLUGIN_PATH="$JRE_HOME/plugin/ns4plugin/:/usr/lib/netscape/plugins"

PAGER="/usr/bin/less"

unset i
```

Tento konfigurační soubor nastavuje některé základní proměnné prostředí a také některé pro
měnné pro uživatele, kteří ve webovém prohlížeči používají Javu či javové aplikace. Viz kapitolu
„Proměnné“.

Informace o podmiňovacím příkazu if naleznete v kapitole „Podmíněné příkazy“. O smyčkách
a příkazu for se více dozvíte v kapitole „Tvorba interaktivních skriptů“.Zdrojový kód bashe se distribuuje i se vzorovými soubory profile. Při nastavování vlastního pro-středí se můžete inspirovat jak těmito příklady, tak výše uvedeným příkladem, vždy je ale budetemuset upravit podle svých potřeb.

/etc/bashrc

Na systémech, kde se používá více shellů, bude rozumnější uložit specifická nastavení bashe právě do tohoto souboru. Soubor /etc/profile se totiž zpracovává i při spouštění jiných shellů, napří-klad Bourne shellu. Oddělením konfiguračních souborů pro jednotlivé typy shellů se předejde chybám, které by jiné shelly generovaly v situacích, kdy nerozumějí syntaxi bashe. V těchto pří-padech uživatelské soubory ~/.bashrc typicky obsahují odkaz na /etc/bashrc, čímž se zaručí provedení tohoto inicializačního souboru.

Další obvyklé uspořádání vypadá tak, že soubor /etc/profile obsahuje pouze nastavení pro-měnných prostředí a spouštěcích parametrů programů, soubor /etc/bashrc obsahuje celosysté-mová nastavení shellových funkcí a aliasů. Na soubor /etc/bashrc se odkazuje buď ze souboru /etc/profile nebo z inicializačních souborů konkrétních uživatelů.
Zdrojová distribuce bashe obsahuje příklady konfiguračních souborů bashrc, naleznete je také v adresáři /usr/share/doc/bash-(verze_bashe)/startup-files. Takto vypadá část souboru bashrc obsaženého v dokumentaci:

```
alias ll='ls -l' alias dir='ls -ba' alias c='clear' alias ls='ls --color'

alias mroe='more' alias pdw='pwd' alias sl='ls --color'
```

```
pskill()
{
local pid


pid=$(ps -ax | grep $1 | grep -v grep | gawk '{ print $1 }')
echo -n "killing $1 (process $pid)..."
kill -9 $pid
echo "slaughtered."

}
```

Kromě obecných aliasů se zde definují také užitečné aliasy, jimiž se „opravují“ některé časté pře-klepy. O aliasech budeme hovořit v kapitole „Aliasy“. Soubor dále definuje funkci pskill, o funk-cích budeme hovořit v kapitole „Funkce“.

Uživatelské konfigurační soubory

Nenašli jste uživatelské konfigurační soubory?

Soubory nemusí být ve vašem domovském adresáři standardně vytvořeny, v takovém pří
padě si je vytvořte sami.

~/.bash_profile

Jde o preferovaný soubor pro individuální konfiguraci uživatelského prostředí. V tomto souboru si uživatel může nastavit specifické konfigurační volby nebo přepsat výchozí nastavení:

```
franky~> cat .bash_profile ######################################################################
#                                                                              #
#    .bash_profile file                                                        #
#                                                                              #
#    Spouští se z bash shellu při přihlášení.                                  #
#                                                                              #
################################################################################

source ~/.bashrc source ~/.bash_login case "$OS" in
IRIX)
stty sane dec
stty erase
    ;; # SunOS) # stty erase # ;;
  *)
    stty sane
    ;;

esac
```

Uživatel si v tomto souboru nastavuje chování klávesy backspace podle toho, k jakému operač-nímu systému se přihlašuje. Kromě toho se načtou soubory ~/.bashrc a ~/.bash_login.

~/.bash_login

Tento soubor obsahuje nastavení, která se za normálních okolností provádějí jen při přihlášení do systému. V následujícím příkladu nastavuje hodnotu umask a vypíše přihlášené uživatele. Kromě toho uživateli zobrazí kalendář aktuálního měsíce:

```
################################################################################ ## Bash_login file ## ## příkazy prováděné bashem při
```

přihlášení ## (voláno z .bash_profile) ## ochrana souborůumask 002 # já a skupina všechno, ostatní jen číst# různéwcal `date +"%m"` `date +"%Y"`

Tento soubor se automaticky načte v případě neexistence souboru ~/.bash_profile.

~/.profile

Jestliže neexistují soubory ~/.bash_profile ani ~/.bash_login, načte se soubor ~/.profile. Může obsahovat stejná konfigurační nastavení, načítají jej nicméně i ostatní shelly. Nezapomeňte, že jiné shelly nemusí syntaxi bashe rozumět.

~/.bashrc

V současné době se běžně používají nepřihlašovací shelly, například pokud v grafickém prostře-dí spustíte okno terminálu. V takovém okně uživatel nezadává jméno a heslo, neprovádí se auten-tizace. Bash v takovém případě načítá soubor ~/.bashrc. Tento soubor bývá volán i z některého ze souborů spouštěných při inicializaci přihlašovacího shellu, aby nebylo nutné zadávat stejná nastavení ve více souborech.

V následujícím příkladu uživatelského souboru .bashrc se nejprve načte celosystémově platný soubor /etc/bashrc a poté se definuje několik aliasů a proměnných:

```
franky ~> cat .bashrc # /home/franky/.bashrc

      # Globální nastavení if [ -f /etc/bashrc ]; then . /etc/bashrc
fi

# volby shellu

set -o noclobber

# moje proměnné

export PS1="\[\033[1;44m\]\u \w\[\033[0m\] "
export PATH="$PATH:~/bin:~/scripts"


# moje aliasy


alias cdrecord='cdrecord -dev 0,0,0 -speed=8'
alias ss='ssh octarine'
alias ll='ls -la'


# úpravy pro mozillu


MOZILLA_FIVE_HOME=/usr/lib/mozilla
LD_LIBRARY_PATH=/usr/lib/mozilla:/usr/lib/mozilla/plugins
MOZ_DIST_BIN=/usr/lib/mozilla
MOZ_PROGRAM=/usr/lib/mozilla/mozilla-bin
export MOZILLA_FIVE_HOME LD_LIBRARY_PATH MOZ_DIST_BIN MOZ_PROGRAM


# úpravy fontů
alias xt='xterm -bg black -fg white &'


# nastavení BitchX
export IRCNAME="frnk"


# KONEC
franky ~>
```

Další příklady naleznete ve zdrojovém balíku bashe. Nezapomeňte, že aby vzorové soubory fun govaly ve vašem prostředí, budete je nejspíš muset upravit.

O aliasech hovoříme v kapitole „Aliasy“.

~/.bash_logout

Tento soubor obsahuje instrukce prováděné při odhlašování. V našem příkladu dojde ke smazání okna terminálu. Je to užitečné při vzdáleném připojování, po odhlášení bude okno smazáno.

```
franky ~> cat .bash_logout ##############################################################################
#                                                                                                      #
#    Bash_logout file                                                                                  #
#                                                                                                      #
#    příkazy prováděné při odhlašování ze shellu                                                       #
#                                                                                                      #

##############################################################################
clear
franky ~>
```

Modifikace konfiguračních souborů shellu

Když kterýkoliv z uvedených souborů změníte, musíte se buď k systému znovu přihlásit nebo spustit nový soubor příkazem source. Při tomto způsobu spuštění se změny promítnou do aktu-ální relace shellu:

Většina shellových skriptů se provádí v privátním prostředí – synovské procesy nedědí proměn-né, pokud je rodičovský shell neexportuje. Volání skriptu příkazem source je metoda, jak změnyprovést v aktuálním prostředí, jak nastavit proměnné v aktuální instanci shellu.

Náš příklad demonstroval použití různých promptů pro různé uživatele. V tomto případě zname
ná červená barva nebezpečí. Je-li prompt zelený, nemusíte být tak moc opatrní.

Jak jsme již uvedli, příkaz source soubor je ekvivalentní zápisu . soubor.Pokud byste se ve všech těch konfiguračních souborech ztratili a zjistili, že se uplatňuje nějakénastavení, jehož původ vám není jasný, použijte příkaz echo stejně jako při ladění skriptů v kapi-tole „Ladění částí skriptu“. Můžete přidat například takovéto řádky:

```
echo "Te provádím .bash_profile..."
```

nebo

```
echo "Nastavuji PS1 v .bashrc:"
export PS1="[něco]"
echo "PS1 má hodnotu $PS1"
```

Proměnné

Typy proměnných

Jak už jsme viděli v předešlých příkladech, zavedenou konvencí je zadávat názvy proměnných shellu velkými písmeny. Bash udržuje seznam dvou typů proměnných:

Globální proměnné

Globální proměnné či též proměnné prostředí jsou k dispozici ve všech shellech. Tyto proměnnémůžete vypsat příkazy env nebo printenv. Oba jsou součástí balíku *sh-utils*.Takto vypadá typický výstup:

```
[jura@jv2 ~]$ printenv SSH_AGENT_PID=2930 HOSTNAME=jv2 DESKTOP_STARTUP_ID= SHELL=/bin/bash TERM=xterm
HISTSIZE=1000 GTK_RC_FILES=/etc/gtk/gtkrc:/home/jura/.gtkrc-1.2-gnome2 WINDOWID=39845962 QTDIR=/usr/lib/qt-3.3 USER=jura
<zkráceno>

[jura@jv2 ~]$
```

Lokální proměnné

Lokální proměnné jsou k dispozici pouze v aktuálním shellu. Vestavěným příkazem set bez dal-ších voleb vypíšete všechny proměnné (včetně proměnných prostředí) a funkce. Výpis bude seřa-zen podle lokalizačního nastavení a zobrazen v použitelném formátu.

Následující příklad vypisuje rozdílový soubor mezi výpisy příkazů printenv a set po odfiltrování funkcí, které set vypisuje taky:

```
[jura@jv2 ~]$ diff set.sorted printenv.sorted | grep "<" | awk '{ print
$2 }'_=BASH_ARGC=()BASH_ARGV=()BASH=/bin/bashBASH_LINENO=()BASH_SOURCE=()BASH_VERSINFO=([0]="3"BASH_VERSI
ON='3.00.15(1)-
release'COLORS=/etc/DIR_COLORS.xtermCOLUMNS=157DIRSTACK=()EUID=500GROUPS=()HISTFILE=/home/jura/.bash_historyHISTFI
LESIZE=1000HOSTTYPE=i686IFS=$'LESSOPEN='|/usr/bin/lesspipe.shLINES=42LS_COLORS='no=00:fi=00:di=00;34:ln=00;36:pi=40;33...
<zkráceno>MACHTYPE=i686-redhat-linux-gnuMAILCHECK=60OPTERR=1
OPTIND=1 OSTYPE=linux-gnu PIPESTATUS=([0]="0" PPID=3794 PROMPT_COMMAND='echo PS1='[\u@\h PS2='> PS4='+
SHELLOPTS=braceexpand:emacs:hashall:histexpand:history:interactive-comments:monitor SUPPORTED=cs_CZ.UTF-8:cs_CZ:cs UID=500
[jura@jv2 ~]$
```

Awk

O programu GNU awk hovoříme v kapitole „Programovací jazyk GNU awk“.

Dělení podle obsahu

Kromě rozdělení na lokální a globální můžeme proměnné rozdělit také do kategorií podle toho, jaký je typ jejich obsahu. Na základě tohoto kritéria se proměnné dělí do čtyř kategorií:

Řetězcové proměnné.
Celočíselné proměnné.
Konstantní proměnné.
Pole.

O jednotlivých typech budeme více hovořit v kapitole „Více o proměnných“. Pro tuto chvíli si vystačíme s celočíselnými a řetězcovými proměnnými.

Vytváření proměnných

V názvech proměnných se rozlišují velká a malá písmena a konvencí je dáno, že se pojmenová-vají velkými písmeny. Občas se pro názvy lokálních proměnných používají malá písmena. Samo-zřejmě můžete názvy volit zcela libovolně a míchat velká i malá písmena. Názvy proměnných mohou obsahovat číslice, nicméně název nesmí číslicí začínat.

```
[jura@jv2 ~]$ export 1number=1 bash: export: `1number=1': not a valid identifier
```

Proměnnou vytvoříte následujícím zápisem:

```
NÁZEV_PROMĚNNÉ="hodnota"
```

Kolem rovnítka nesmí být mezery, to by vedlo k chybě. Pokud do proměnné přiřazujete řetězco-vou hodnotu, je dobrým zvykem zapisovat ji do uvozovek, eliminuje se tím riziko vzniku chyb. Několik příkladů používajících velká a malá písmena, čísla a mezery:

```
franky ~> MYVAR1="2"

franky ~> echo $MYVAR1 franky ~> first_name="Franky"

franky ~> echo $first_name Franky

franky ~> full_name="Franky M. Singh"

franky ~> echo $full_name Franky M. Singh

franky ~> MYVAR-2="2" bash: MYVAR-2=2: command not found

franky ~> MYVAR1 ="2" bash: MYVAR1: command not found

franky ~> MYVAR1= "2" bash: 2: command not found
```

franky ~> unset MYVAR1 first_name full_name

franky ~> echo $MYVAR1 $first_name $full_name <--žádný výstup-->

franky ~>

Export proměnných

Proměnné vytvořené postupem uvedeným v předchozím příkladu budou k dispozici pouze v aktuálním shellu. Jsou to lokální proměnné – synovské procesy aktuálního shellu je nebudou znát. Chcete-li proměnnou předat subshellu, musíte ji *exportovat* vestavěným příkazem export. Jakmile proměnnou exportujete, stává se z ní proměnná prostředí, tedy globální proměnná. Nastavení a export proměnné se typicky provádí najednou:

export NÁZEV_PROMĚNNÉ="*hodnota*"

Subshell může proměnné zděděné od rodiče modifikovat, provedené změny se ovšem v rodičovském procesu nepromítnou. Ukazuje to následující příklad:

franky ~> full_name=*"Franky M. Singh"*

franky ~> bash

franky ~> echo $full_name

franky ~> exit

franky ~> export full_name

franky ~> bash

franky ~> echo $full_name Franky M. Singh
franky ~> export full_name=*"Charles the Great"*

franky ~> echo $full_name Charles the Great

franky ~> exit

franky ~> echo $full_name Franky M. Singh

franky ~>

Když jsme zkusili poprvé v subshellu vypsat proměnnou full_name, neexistovala (příkaz echo vypsal prázdný řetězec). Subshell jsme ukončili a v rodičovském procesu jsme provedli export proměnné full_name. (V tomto případě jsme proměnnou exportovali odděleně od jejího nasta-vení.) V dalším spuštěném subshellu už byla tato proměnná viditelná. Změnili jsme její obsah, nic-méně v rodičovském procesu zůstal obsah původní.

Rezervované proměnné Rezervované proměnné Bourne shellu

Bash používá některé proměnné stejně jako Bourne shell. V některých případech těmto proměn-ným bash přiřazuje hodnoty. Následující tabulka uvádí přehled těchto základních rezervovaných proměnných.

Název proměnné	Definice
CDPATH	Dvojtečkami oddělený seznam adresářů používaný jako prohledávací cesta vestavěného příkazu .
HOME	Domovský adresář aktuálně přihlášeného uživatele, implicitní cíl příkazu . Tato hodnota se používá rovněž k expanzi tildy.
IFS	Seznam oddělovačů polí, používá se v případech, kdy shell v rámci expanze odděluje slova.
MAIL	Je-li v této proměnné uložen název souboru a není-li nastavena proměnná MAILPATH, bash uživatele informuje o přijetí pošty v daném souboru.
MAILPATH	Dvojtečkami oddělený seznam názvů souborů, ve kterých shell pravidelně hledá novou poštu.
OPTARG	Hodnota posledního parametru zpracovaného vestavěným příkazem .

OPTIND	Index posledního parametru zpracovaného vestavěným příkazem .
PATH	Dvojtečkami oddělený seznam adresářů, v nichž shell hledá příkazy.
PS1	Primární prompt. Výchozí hodnota je „\s-\v\$ “.
PS2	Sekundární prompt. Výchozí hodnota je „> “.

Rezervované proměnné Bourne shellu

Rezervované proměnné bashe

Následující proměnné nastavuje nebo používá bash, pro jiné shelly obvykle nemají zvláštní význam.

Proměnná	Definice
auto_resume	Tato proměnná určuje, jak shell interaguje s uživatelem a řízením úloh.
BASH	Plná cesta použitá ke spuštění aktuální instance bashe.
BASH_ENV	Je-li tato proměnná nastavena při spuštění bashe za účelem provedení skriptu, její hodnota se expanduje a použije jako název inicializačního souboru, který se zpracuje před samotným skriptem.
BASH_VERSION	Číslo verze aktuální instance bashe.
BASH_VERSINFO	Pole jen pro čtení, prvky obsahují informace o verzi aktuální instance bashe.
COLUMNS	Používá se vestavěným příkazem ke zjištění šířky terminálového okna při výpisu do sloupců. Nastavuje se automaticky po přijetí signálu SIGWINCH.
COMP_CWORD	Index slova obsahujícího aktuální pozici kurzoru v poli ${COMP_WORDS}.
COMP_LINE	Aktuální příkazový řádek.
COMP_POINT	Index aktuální pozice kurzoru relativně k začátku aktuálního příkazu.
COMP_WORDS	Pole obsahující jednotlivá slova na aktuálním příkazovém řádku.
COMPREPLY	Pole, z nějž bash čte možná dokončení generovaná shellovou funkcí spouštěnou programova-telným ukončovaním.
DIRSTACK	Pole obsahující adresářový zásobník.
EUID	Číselná hodnota efektivního uživatelského ID aktuálního uživatele.
FCEDIT	Výchozí editor používaný volbou -evestavěného příkazu .
FIGNORE	Dvojtečkami oddělený seznam suffixů ignorovaných při dokončování názvů souborů.
FUNCNAME	Název aktuálně prováděné funkce shellu.
GLOBIGNORE	Dvojtečkami oddělený seznam šablon definujících názvy souborů ignorovaných při expanzi názvů souborů.
GROUPS	Pole obsahující seznam skupin, jichž je aktuální uživatel členem.
histchars	Až trojice znaků řídící expanzi historie, rychlou substituci a tokenizaci.
HISTCMD	Index aktuálního příkazu v seznamu historie.
HISTCONTROL	Definuje, zda má být příkaz přidán do souboru historie.
HISTFILE	Název souboru, do nějž se historie zapisuje. Výchozí nastavení je ~/.bash_history.
HISTFILESIZE	Maximální počet řádků v souboru historie, výchozí hodnota je 500.
HISTIGNORE	Dvojtečkami oddělený seznam vzorů udávající, jaké příkazy mají být zapisovány do historie.
HISTSIZE	Maximální počet příkazů zachovávaný v seznamu historie příkazů, výchozí hodnota je 500.
HOSTFILE	Obsahuje název souboru ve stejném formátu, jako je /etc/hosts, z nějž bude shell číst, když potřebuje doplnit název počítače.
HOSTNAME	Název tohoto počítače.
HOSTTYPE	Řetězec popisující počítač, na němž bash běží.
IGNOREEOF	Řídí reakci shellu na znak EOF přijatý na vstupu.
INPUTRC	Název inicializačního souboru knihovny Readline, standardně /etc/inputrc.
LANG	Lokalizační nastavení pro kategorie, které nejsou specificky definovány v proměnných začínají-cích na LC_.
LC_ALL	Tato proměnná přepisuje hodnotu proměnné LANGa všech ostatních proměnných LC_defi-nujících různé nastavení lokalizačních kategorií.

Proměnná	Definice
LC_COLLATE	Tato proměnná řídí porovnávací pořadí používané při řazení výsledků expanze názvů souborů a dále ovlivňuje chování definic rozsahů, tříd ekvivalence a porovnávacích sekvencí při expanzi souborů a porovnávání se vzory.

LC_CTYPE	Tato proměnná řídí interpretaci znaků a chování tříd znaků při expanzi souborů a porovnávání se vzory.
LC_MESSAGES	Tato proměnná definuje lokalizační nastavení používané při překladu řetězců ve dvojitých uvozovkách uvedených symbolem „$“.
LC_NUMERIC	Tato proměnná udává lokalizační nastavení používané pro formátování čísel.
LINENO	Číslo právě zpracovávaného řádku ve funkci nebo skriptu.
LINES	Používá se vestavěným příkazem k určení výšky sloupců při výpisu ve sloupcích.
MACHTYPE	Řetězec s úplným popisem typu systému, na němž bash běží, ve standardním formátu GNU CPU-COMPANY-SYSTEM.
MAILCHECK	Jak často (v sekundách) má shell kontrolovat novou poštu v souborech zadaných v proměnných MAILPATHnebo MAIL.
OLDPWD	Předchozí aktuální adresář nastavený vestavěným příkazem .
OPTERR	Má-li tato proměnná hodnotu 1, bude bash vypisovat chybová hlášení generovaná vestavěným příkazem .
OSTYPE	Řetězec popisující operační systém, na němž bash běží.
PIPESTATUS	Pole obsahující seznam návratových hodnot procesů v poslední vykonávané sekvenci rour (která může obsahovat jediný příkaz).
POSIXLY_CORRECT	Pokud tato proměnná existuje při startu , shell bude pracovat v režimu POSIX.
PPID	ID rodičovského procesu shellu.
PROMPT_COMMAND	Pokud je tato proměnná nastavena, její hodnota je interpretována jako příkaz, který se provede před každým vypsáním primárního promptu (PS1).
PS3	Hodnota této proměnné se používá jako prompt příkazu . Výchozí hodnota je „#?“.
PS4	Tento prompt se vypisuje před vypsáním příkazového řádku, je-li nastavena volba -xshellu. Výchozí hodnota je „+“.
PWD	Aktuální pracovní adresář nastavený příkazem .
RANDOM	Při každém čtení této proměnné je vráceno náhodně číslo mezi 0 a 32 767. Zápisem hodnoty do této proměnné dojde k inicializaci generátoru.
REPLY	Výchozí proměnná pro vestavěný příkaz .
SECONDS	Tato proměnná obsahuje počet sekund od spuštění shellu.
SHELLOPTS	Dvojtečkami oddělený seznam zapnutých voleb shellu.
SHLVL	Inkrementuje se při každém spuštění nové instance bashe.
TIMEFORMAT	Tato proměnná obsahuje formátovací řetězec hodnoty příkazu při jeho předávání rourou.
TMOUT	Je-li tato hodnota větší než nula, představuje výchozí timeout příkazu . V interaktivním shellu je chápána jako počet sekund, po něž se má čekat na zadání vstupu od zobrazení primárního promptu. Nebude-li v požadovaném čase zadán žádný vstup, bash se ukončí.
UID	Číselná hodnota reálného uživatelského ID aktuálního uživatele.

Rezervované proměnné bashe

Podrobnější informace naleznete na manuálových a informačních stránkách bashe a v dokumen-

taci. Některé proměnné slouží jen pro čtení, některé se nastavují automaticky, u některých změna jejich hodnoty způsobí, že proměnná přestává dávat smysl.

Speciální proměnné

Některé proměnné interpretuje shell speciálním způsobem. Jejich hodnoty je možné pouze číst, není možné do nich přiřazovat.

Znak	Definice
*	Expanduje se na poziční parametr počínaje od jedné. Dochází-li k expanzi ve dvojitých uvozovkách, expanduje se na jediné slovo s hodnotami jednotlivých parametrů oddělenými prvním znakem speciální proměnné IFS.
@	Expanduje se na poziční parametr počínaje od jedné. Dochází-li k expanzi ve dvojitých uvozovkách, každý parametr je expandován na samostatné slovo. # Expanduje se na počet pozičních parametrů (dekadicky). ? Expanduje se na návratový kód poslední na popředí spuštěné sekvence příkazů.
-	Expanduje se na aktuálně nastavené volby shellu, a už zadané při spuštění, nastavené příkazem nebo nastavené samotným shellem (například -i). $ Expanduje se na PID shellu. ! Expanduje se na PID posledního na pozadí spuštěného (asynchronního) příkazu. Expanduje se na název shellu nebo skriptu.
_	Nastavuje se při spuštění shellu a obsahuje absolutní cestu k shellu či prováděnému skriptu. Poté se expanduje na poslední parametr předchozího příkazu, po expanzi. Při spouštění každé-ho příkazu se nastaví na plnou cestu k příkazu a exportuje se do prostředí příkazu. Při kontrole pošty obsahuje název souboru s poštou.

Speciální proměnné bashe

Poziční parametry jsou slova uvedená za názvem skriptu (tedy parametry skriptu). Předávají se v proměnných $1, $2, $3 a tak dále. Dokud je to potřeba, přidávají se proměnné do interního pole. Proměnná $# obsahuje celkový počet parametrů. Demonstruje to následující jednoduchý skript:

```
#!/bin/bash


# positional.sh
# Skript přečte a vypíše tři poziční parametry


POSPAR1="$1"
POSPAR2="$2"
POSPAR3="$3"


echo "$1 je první poziční parametr, \$1."
echo "$2 je druhý poziční parametr, \$2."
echo "$3 je třetí poziční parametr, \$3."
echo
echo "Celkem je definováno $# pozičních parametrů."
```

Při spuštění je možné zadat libovolný počet parametrů:

```
[jura@jv scripts]$ ./positional.sh raz dva tři čtyři pětraz je první poziční parametr, $1.dva je druhý poziční parametr, $2.tři je třetí poziční parametr, $3.

Celkem je definováno 5 pozičních parametrů.


[jura@jv scripts]$ ./positional.sh raz dvaraz je první poziční parametr, $1.dva je druhý
```

poziční parametr, $2.
je třetí poziční parametr, $3.

Celkem je definováno 2 pozičních parametrů. [jura@jv scripts]$

Další informace o vyhodnocování parametrů naleznete v kapitole „Podmíněné příkazy“ a „Smyč-ka while“.Ukažme si několik příkladů práce s dalšími speciálními parametry:

```
[jura@jv2 ~]$ grep dictionary /usr/share/dict/words antidictionary benedictionary dictionary dictionary's dictionary-proof extradictionary nondictionary

[jura@jv2 ~]$ echo $_ /usr/share/dict/words

[jura@jv2 ~]$ echo $$ 4222

[jura@jv2 ~]$ firefox &
[1] 4241

[jura@jv2 ~]$ echo $! 4241

[jura@jv2 ~]$ echo $0 bash

[jura@jv2 ~]$ echo $? 0

[jura@jv2 ~]$ ls nejakynesmysl ls: nejakynesmysl: není souborem ani adresářem

[jura@jv2 ~]$ echo $? 1

[jura@jv2 ~]$
```

Uživatel nejprve zadal příkaz grep, což vedlo k nastavení proměnné _. ID procesu spuštěného shellu je 4222. Po spuštění úlohy na pozadí obsahuje ! PID procesu na pozadí. Spuštěným shel-lem je bash. Když při provádění příkazu dojde k chybě, proměnná ? obsahuje nenulový návra-tový kód.

Recyklace skriptů pomocí proměnných

Kromě toho, že proměnné zvyšují čitelnost skriptu, umožní vám také snáze skript upravit pro jiné prostředí nebo pro jiný účel. Vezměme si příklad následujícího jednoduchého skriptu, který zálo-huje domovský adresář uživatele *franky* na vzdálený server:

```
#!/bin/bash

# Tento skript zálohuje můj domovský adresář.

cd /home

# Vytvoříme archiv tar cf /var/tmp/home_franky.tar franky > /dev/null 2>&1

# Nejprve odstraníme starý soubor bzip2. Chyby přesměrováváme, protože pokud by archiv neexistoval # zbytečně by se zobrazovaly. Pak vytvoříme nový komprimovaný soubor. rm /var/tmp/home_franky.tar.bz2 2> /dev/null bzip2 /var/tmp/home_franky.tar

# Zkopírujeme soubor na cílový server - máme dobře nastavené klíče, takže to funguje samo. scp /var/tmp/home_franky.tar.bz2 bordeaux:/opt/backup/franky > /dev/null 2>&1

# Do logu zapíšeme zprávu o provedení zálohy date > /home/franky/log/home_backup.log echo zálohováno! > /home/franky/log/home_backup.log
```

První problém spočívá v tom, že pokud názvy souborů a adresářů pokaždé zadáváte ručně, snad-no můžete udělat chybu. No a za druhé, *franky* může chtít svůj skript po čase věnovat *carol* ata bude muset skript na různých místech upravovat tak, aby zálohoval její domovský adresář. To samé platí v případě, že by *franky* chtěl skriptem zálohovat něco jiného. Kvůli snazší recyklaci skriptů se doporučuje názvy všech souborů, adresářů, uživatelská jména, názvy serverů a podob-ně definovat jako proměnné. Pak vám při každé úpravě postačí změnit hodnotu na jednom jedi-ném místě bez nutnosti skript procházet a zkoumat, kde se hodnota používá. Takto vypadá vylep-šený příklad:

```
#!/bin/bash

# Tento skript zálohuje můj domovský adresář.
```

```
# Upravte si hodnoty proměnných podle svých potřeb: BACKUPDIR=/home BACKUPFILES=franky TARFILE=/var/tmp/home_franky.tar
BZIPFILE=/var/tmp/home_franky.tar.bz2 SERVER=bordeaux REMOTEDIR=/opt/backup/franky
LOGFILE=/home/franky/log/home_backup.log

cd $BACKUPDIR

# Vytvoříme archiv tar cf $TARFILE $BACKUPFILES > /dev/null 2>&1

# Nejprve odstraníme starý soubor bzip2. Chyby přesměrováváme, protože pokud by archiv neexistoval # zbytečně by se zobrazovaly. Pak
vytvoříme nový komprimovaný soubor. rm $BZIPFILE 2> /dev/null bzip2 $TARFILE

# Zkopírujeme soubor na cílový server - máme dobře nastavené klíče, takže to funguje samo. scp $BZIPFILE $SERVER:$REMOTEDIR
> /dev/null 2>&1

# Do logu zapíšeme zprávu o provedení zálohy date > $LOGFILE echo zálohováno! > $LOGFILE
```

> Velké adresáře a pomalá linka
>
> Výše uvedený skript je pouhý snadno pochopitelný příklad, zálohujeme malý adresář přes lokální síť. V závislosti na rychlosti připojení, velikosti zálohovaných dat a umístění zálož-ního serveru by tento zálohovací mechanismus mohl běžet velice dlouho. Při zálohování velkých adresářů přes pomalé linky doporučujeme obsah adresářů synchronizovat příka-zem rsync.

Uvozování znaků

Proč?

Řada znaků a slov má v tom či onom kontextu speciální význam. Uvozování slouží k odstranění speciálního významu znaků nebo slov: Uvození může vypnout speciální obsluhu speciálních znaků, může zabránit v rozeznání rezervovaných slov a může vypnout expanzi parametrů.

Escapování znaků

Escapováním se potlačuje speciální význam jednoho znaku. V bashi se jako escapovací znak pou-žívá neuvozené obrácené lomítko (\). Tím dojde k zachování literálního významu bezprostředně následujícího znaku, s výjimkou znaku nového řádku. Následuje-li za obráceným lomítkem znak nového řádku, znamená to pokračování na dalším řádku terminálu – obrácené lomítko bude ze vstupního řetězce odstraněno a efektivně se ignoruje.

```
franky ~> date=20021226

franky ~> echo $date 20021226

franky ~> echo \$date $date
```

V tomto příkladu jsme vytvořili proměnnou date a uložili jsme do ní nějakou hodnotu. První pří-kaz echo vypíše její hodnotu, ale ve druhém případě je význam znaku dolar potlačen, a nedojde tudíž k expanzi parametru a výpisu obsahu proměnné date.

Apostrofy

Apostrofy (') zachovávají literální význam všech znaků mezi apostrofy. Mezi apostrofy není možné zapsat další apostrof, ani pokud jej uvedete obráceným lomítkem.
Pokračujme v předchozím příkladu:

```
franky ~> echo '$date' $date
```

Uvozovky

Při použití uvozovek (“) se zachovává literální význam všech znaků kromě znaku dolar, obrácených apostrofů (`) a obráceného lomítka.
Uvnitř uvozovek si dolar i obrácené apostrofy zachovávají svůj speciální význam. Obrácené lomítko si zachovává svůj význam, jen pokud za ním následuje dolar, obrácený apo-strof, uvozovka, obrácené lomítko nebo znak nového řádku. Z řetězce v uvozovkách bude obrá-cené lomítko odstraněno, právě když za ním následuje jeden z uvedených znaků. Pokud za obrá-ceným lomítkem následuje jiný znak bez speciálního významu, zůstane obrácené lomítko v řetěz-ci a bude shellem zpracováno.

franky ~> echo *"$date"*20021226

franky ~> echo *"`date`"*Sun Apr 20 11:22:06 CEST 2003

franky ~> echo *"Řek' bych: \"Udělej to!\""*Řek' bych: "Udělej to!"

franky ~> echo *"\"*> <--- čeká se na pokračování vstupu --->

franky ~> echo *"\\"*
\

Uvozování podle ANSI-C

Slova ve tvaru $'*řetězec*' jsou chápána speciálním způsobem. Toto slovo bude expandováno na zadaný řetězec, přičemž obráceným lomítkem escapované znaky budou nahrazeny speciálními znaky dle standardu ANSI-C. Tyto speciální escapovací sekvence naleznete v dokumentaci k bashi. Malá ukázka:

[jura@jv2 ~]$ echo $'cosi1\ncosi2' cosi1 cosi2

Lokalizace

Řetězec v uvozovkách, jimž předchází znak dolaru, bude přeložen podle aktuálního nastavení lokalizace (viz proměnnou LANG). Pokud je nastavena lokalizace „C“ nebo „POSIX“, dolar se igno-ruje. Dojde-li k překladu a náhradě, nový text bude uzavřen v uvozovkách.

Expanze shellu

Obecně

Poté co je příkaz rozdělen na *tokeny – slova* (viz kapitolu „Syntaxe shellu“), tyto tokeny se expan-dují. Existuje osm typů prováděných expanzí, budeme o nich hovořit v následujících částech v tom pořadí, v jakém se expanze provádějí.

Po provedení všech expanzí dochází k odstranění uvozovacích znaků.

Expanze složených závorek

Expanze složených závorek je mechanismus, jímž mohou být generovány libovolné řetězce. Řetě-zec podléhající expanzi je tvořen nepovinnou *preambulí*, následuje libovolný počet čárkami oddě-lených řetězců uzavřených ve složených závorkách a nakonec nepovinný *postscript*. Celý výraz se expanduje na řetězce ve složených závorkách, z nichž každý bude začínat preambulí a končit postscriptem, v tom pořadí, jak byly v závorkách uvedeny.

Složené závorky je možné vnořovat:

[jura@jv ~]$ echo p{lá,řá,lo}t plát přát plot

[jura@jv ~]$ echo p{{l,ř,s}á,lo}t plát přát psát plot

[jura@jv ~]$ echo {{1,2,3}{0,1,2,3,4,5,6,7,8,9}} 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39

Expanze složených závorek se provádí před všemi ostatními expanzemi, případné znaky se spe-ciálním významem pro následující expanze zůstanou zachovány. Jde o striktně textový mechanis-mus. Obsah expanze ani text mezi závorkami nepodléhá žádné syntaktické interpretaci bashe. Aby se předešlo konfliktům s expanzí parametrů, řetězec *${* je chápán jako nevhodný pro expanzi slo-žených závorek.

Správně naformátovaný výraz pro expanzi musí obsahovat neuvozenou otevírací a uzavírací závorku a přinejmenším jednu neuvozenou čárku. Nesprávně formátovaný výraz nebude expan-dován a zůstane beze změny.

Expanze tildy

Pokud slovo začíná neuvozenou tildou (~), pak jsou všechny následující znaky až po první neu-vozené lomítko (anebo všechny následující znaky, pokud ve slově neuvozené lomítko není) chá-pány jako *tilde-prefix*. Není-li žádný ze znaků v prefixu uvozen, jsou znaky v prefixu chápány jako přihlašovací jméno uživatele. Je-li přihlašovací jméno prázdné, je tilda nahrazena hodnotou pro-měnné HOME. Není-li proměnná HOME nastavena, použije se domovský adresář uživatele, který spustil shell. Jinak bude prefix nahrazen domovským adresářem uživatele se zadaným přihlašovacím jménem.

[jura@jv ~]$ echo ~ /home/jura [jura@jv ~]$ echo ~jura /home/jura [jura@jv ~]$ echo ~martina /home/martina

Je-li hodnota prefixu ~+, bude prefix nahrazen hodnotou proměnné PWD. Je-li prefix ~-, bude nahrazen hodnotou proměnné OLDPWD, je-li tato nastavena.

[jura@jv ~]$ cd tldp [jura@jv tldp]$ cd ../scripts/ [jura@jv scripts]$ echo ~+ /home/jura/scripts [jura@jv scripts]$ echo ~-/home/jura/tldp

Pokud je za tildou uvedeno číslo (a před ním nepovinně znak + či -), expanduje se výraz na odpo-vídající prvek ze zásobníku adresářů, stejně jako byste volali vestavěný příkaz dirs s parametremshodujícím se se znaky za tildou. Pokud je za tildou uvedeno číslo bez symbolu + nebo -, je tochápáno, jako by byl uveden symbol +.

Jestliže je uvedeno neplatné přihlašovací jméno nebo selže expanze adresářového zásobníku,

zůstane slovo nezměněno.Při přiřazování hodnot proměnných se testuje výskyt neuvozené tildy bezprostředně za znaky :a =. I v těchto případech se provede expanze tildy. Při nastavování hodnot proměnných, jako jsounapříklad PATH, MAILPATH nebo CDPATH, je tedy možné zadávat cesty s použitím tildy, shell správ-ně provede expanzi.

Příklad:

franky ~> export PATH=*"$PATH:~/testdir"*

Řetězec ~/testdir bude expandován na $HOME/testdir, takže má-li $HOME hodnotu /var/home/franky, bude do proměnné PATH přidán adresář /var/home/franky/testdir.

Expanze parametrů shellu a proměnných

Znak *$* uvozuje expanzi parametru, substituci příkazu nebo aritmetickou expanzi. Název či sym-bol expandovaného parametru může být nepovinně uzavřen ve složených závorkách. Ty oddělu-jí název od případných bezprostředně následujících znaků, které už nemají být interpretovány jako součást názvu.

Při použití složených závorek je jako uzavírací závorka chápán první symbol }, který není esca-pován obráceným lomítkem, není uzavřen v uvozeném řetězci a není součástí vložené aritmetic-ké expanze, substituce příkazu nebo expanze parametru.

Základní tvar expanze parametru je *${PARAMETR}*. Dojde k substituci hodnotou parametru. Slo-žené závorky jsou nutné v případě, že parametrem je poziční parametr definovaný více než jed-nou číslicí nebo bezprostředně následovaný dalšími znaky, které už nejsou součástí názvu para-metru.

Je-li prvním znakem názvu parametru vykřičník, dochází k *nepřímé expanzi*. Zbytek znaků za vykřičníkem je chápán jako název proměnné, jejíž hodnota obsahuje název proměnné, jejíž hod-
nota bude výsledkem expanze. V případě neuvedení vykřičníku je výsledkem expanze přímo

hodnota proměnné se zadaným názvem.
Jednoduchou přímou expanzi bezpochyby dobře znáte, k té dochází velmi často už v nejjedno
dušších případech, například:

[jura@jv ~]$ echo $SHELL /bin/bash

Příklad nepřímé expanze může vypadat například takto:

[jura@jv ~]$ PROSTREDI=SHELL [jura@jv ~]$ echo ${PROSTREDI} SHELL [jura@jv ~]$ echo ${!PROSTREDI} /bin/bash
anebo takto

[jura@jv ~]$ echo ${!S*} SECONDS SHELL SHELLOPTS SHLVL SSH_ASKPASS SSH_AUTH_SOCK SSH_CLIENT SSH_CONNECTION SSH_TTY SUPPORTED

Pozor, je to něco jiného než echo $S*, viz následující příklad:

[jura@jv scripts]$ ls commented-script1.sh positional.sh script1.sh [jura@jv scripts]$ echo * commented-script1.sh positional.sh script1.sh

No a protože proměnná S není (s největší pravděpodobností) definována, expanduje se $S na prázdný řetězec a tím pádem stejně jako výše:

[jura@jv scripts]$ echo $S* commented-script1.sh positional.sh script1.sh

Následující konstrukce umožňuje proměnnou použít a zároveň ji vytvořit v případě, že ještě neexistuje:

${PROMĚNNÁ:=*hodnota*}

Například:

[jura@jv ~]$ echo $COSI

[jura@jv ~]$ echo ${COSI:=prostě cosi}prostě cosi[jura@jv ~]$ echo ${COSI:=prostě něco jiného}prostě cosi

Tímto způsobem nicméně nelze přiřazovat speciální parametry, například poziční parametry.
O použití složených závorek při manipulaci s proměnnými budeme ještě hovořit v kapitole „Více

o proměnných“. Další informace můžete nalézt na informačních stránkách bashe.

Substituce příkazů

Při substituci příkazů je příkaz nahrazen svým vlastním výstupem. K substituci dojde, zapíšete-li příkaz takto:

$(příkaz)
anebo pomocí obrácených apostrofů:

`příkaz`

Bash provede expanzi tak, že spustí příkaz a jeho standardním výstupem nahradí substituovaný zápis, přičemž dojde k odstranění všech koncových oddělovačů nového řádku. Oddělovače řádků v těle výstupu odstraněny nebudou, k jejich odstranění však může dojít při dělení slov.

franky ~> echo *`date`* Thu Feb 6 10:06:20 CET 2003

Při použití staršího způsobu zápisu se zpětnými apostrofy si obrácená lomítka zachovávají svůj literální význam kromě situací, kdy za nimi následují znaky $, ` nebo \. První zpětný apostrof, jemuž nepředchází obrácené lomítko, ukončuje substituci příkazu. Pokud použijete zápis $(pří kaz), zachovávají si svůj význam všechny znaky mezi závorkami, žádný není obsluhován speciál ně.

Substituce příkazů je možné vnořovat. Používáte-li zpětné apostrofy, musíte vnitřní dvojici esca

povat pomocí obrácených lomítek.
Pokud se substituce provádí mezi uvozovkami, nad výsledkem se neprovede dělení slov a expan ze názvů souborů.

Aritmetická expanze

Aritmetická expanze umožňuje vyhodnotit aritmetický výraz a provést substituci jeho výsledkem. Aritmetická substituce má následující formát:

$((výraz))

Výraz je chápán stejně, jako by byl uzavřen v uvozovkách, uvozovky uvnitř závorek však nejsou speciálně ošetřovány. Všechny tokeny ve výrazu prochází expanzí parametrů, substitucí příkazů a odstraněním uvození. Aritmetické substituce je možné vnořovat.

Vyhodnocování aritmetických výrazů se provádí celočíselně, s pevnou šířkou hodnot, neprovádí se kontrola přetečení. Dělení nulou je nicméně zachyceno a rozeznáno jako chyba. Operátory jsou přibližně stejné jako v jazyce C. Seřazen podle klesající priority vypadá jejich seznam takto:

Operátor	Význam
a	postinkrementace a postdekrementace proměnné
a	preinkrementace a predekrementace proměnné
a	unární plus a minus
a	logická a binární negace
	umocnění
, a	násobení, dělení, zbytek
a	sčítání, odčítání

a	binární posuv vlevo a vpravo
, , a	operátory porovnání
a	rovnost a nerovnost
	binární AND
	binární XOR

Operátor Význam

binární OR logický AND logický OR podmíněné vyhodnocení , , , , , , přiřazení

, , , a oddělovač mezi výrazy

Aritmetické operátory

Jako operandy je možné použít proměnné shellu, k expanzi parametrů dochází před vyhodnocením výrazu. V rámci výrazu je také možné odkazovat se na proměnné shellu názvem, bez použi-tí syntaxe pro expanzi parametrů. V takovém případě je proměnná vyhodnocena jako aritmetický výraz ve chvíli, kdy je na ni odkazováno. Proměnné je možné ve výrazech použít, aniž by měly nastaveny příznak celočíselnosti.

Konstanty uvozené nulou (*0*) jsou interpretovány jako osmičkové hodnoty. Uvození znaky *0x* nebo *0X* znamená šestnáctkové hodnoty. Obecně je možné použít zápis *základ#číslo*, kde základ může být 2 až 64. Není-li základ definován, pracuje se automaticky se základem 10. Číslice větší než 9 se zapisují malými písmeny, velkými písmeny a znaky @ a _, v tomto pořadí. Je-li základ menší nebo roven 36, je možné číslice 10 až 35 zapisovat jak malými, tak i velkými písmeny.

Operátory se vyhodnocují s respektováním priority. Výrazy v závorkách se vyhodnocují před nostně a přepisují tak prioritní pořadí ve výše uvedené tabulce.Kdykoliv je to možné, měli byste aritmetickou expanzi zapisovat pomocí syntaxe s hranatýmizávorkami:

$[výraz]

V tomto případě se však pouze vypočítá hodnota výrazu, neprovádějí se žádné testy:

franky ~> echo *$[365*24]* 8760

Praktické příklady použití ve skriptech naleznete mimo jiné v kapitole „Porovnání číselných hod-not“.

Substituce procesů

Substituce procesů je podporována na systémech, které podporují pojmenované roury (FIFO) nebo které pojmenovávají otevřené soubory mechanismem /dev/fd. Zapisuje se takto:

<(list)

nebo

>(list)

Proces list bude spuštěn se vstupem nebo výstupem připojeným k FIFO nebo k některému sou-boru v /dev/fd. Název souboru je předán aktuálnímu příkazu jako výsledek expanze. Při použi-tí zápisu *>(LIST)* poskytuje zápis do souboru vstup příkazu list. Při použití zápisu *<(list)* je
možné ze souboru číst výstup příkazu list. Mezi znaky < či > a levou závorkou nesmí být meze

ra, jinak bude zápis chápán jako přesměrování.Substituce procesů se provádí současně s expanzí parametrů a proměnných, substitucí příkazůa aritmetickou expanzí.
Více informací naleznete v kapitole „Přesměrování a deskriptory souborů“.

Dělení slov

Výsledky expanze parametrů, substituce příkazů a aritmetické substituce, pokud nejsou zapsány v uvozovkách, shell dále zpracovává mechanismem dělení slov. Každý znak uvedený v proměnné IFS chápe shell jako oddělovač, a výsledky substitucí pak na těchto znacích rozděluje na samostatná slova. Není-li proměnná IFS nastavena nebo obsahuje-li přesně výchozí hodnotu „<mezera><tabulátor><nový_řádek>“, slouží jako oddělovače slov právě tato trojice znaků. Má-li proměnná IFS jinou než výchozí hodnotu, ignorují se prázdné znaky <mezera> a <ulátor> na začátku a konci slov jen tehdy, jsou-li tyto znaky v proměnné nastaveny. Jako oddělovače slov pak slouží všechny neprázdné znaky definované v proměnné IFS a s nimi bezprostředně sousedící případné prázdné znaky. Je-li hodnota proměnné IFS prázdná, k dělení slov nedochází.

Explicitně zadané prázdné hodnoty ("" nebo '') se zachovávají. Neuvozené prázdné hodnoty, vzniklé jako výsledek expanze parametrů bez nastavené hodnoty, se odstraňují. Pokud ovšem dojde k expanzi s prázdným výsledkem v uvozovkách, prázdná hodnota se zachovává.

> Expanze a dělení slov
>
> Nedošlo-li k expanzi, neprovádí se dělení slov.

Expanze názvů souborů

Po rozdělení slov a pokud nebyla zadána volba -f (viz kapitolu „Ladění částí skriptu“) následně bash v každém slově hledá znaky *, ? a [. Pokud některý z těchto znaků nalezne, je slovo chápá-no jako *vzor* a nahradí se abecedně seřazenými názvy všech souborů, které vzoru vyhovují. Nejsou-li nalezeny žádné vyhovující soubory a je vypnuta volba shellu nullglob, zůstane slovo nezměněno. Pokud je volba nullglob zapnuta a nejsou nalezeny žádné soubory, bude slovo odstraněno. Je-li zapnuta volba nocaseglob, vyhledávání souborů se provádí bez ohledu na velká a malá písmena.

Je-li vzor použit ke generování názvu souboru a není-li nastavena volba dotglob, musí být tečka na začátku názvu souboru nebo bezprostředně za lomítkem vždy zadána explicitně. Explicitně musí být vždy zadáno lomítko. V jiných případech není tečka ošetřována speciálně.

Pomocí proměnné GLOBIGNORE je možno omezit množinu názvů, které budou brány jako vyho-vující vzoru. Je-li proměnná GLOBIGNORE nastavena, budou z výsledného seznamu odstraněny všechny soubory, které vyhovují zadanému vzoru a zároveň vyhovují některému ze vzorů v pro-měnné GLOBIGNORE. Je-li proměnná GLOBIGNORE nastavena a je neprázdná, budou navíc vždy ignorovány názvy . a ..(dvě tečky). Nastavení proměnné GLOBIGNORE na neprázdnou hodnotu však automaticky zapíná volbu dotglob, budou tedy nalezeny všechny soubory, jejichž název začíná tečkou. Chcete-li zachovat původní chování, kdy se soubory začínající tečkou ignorují, nastavte jako jeden ze vzorů v GLOBIGNORE vzor ".*". Při zrušení proměnné GLOBIGNORE dojde k vypnutí volby dotglob.

Aliasy

Co jsou to aliasy?

Alias umožňuje nahrazení slova řetězcem v případě, že je dané slovo použito jako první slovo jed-noduchého příkazu. Shell udržuje seznam aliasů, které je možno nastavovat a rušit vestavěnými příkazy alias a unalias. Zadáte-li příkaz alias bez dalších parametrů, zobrazí se seznam všech aktuálně nastavených aliasů:

```
franky: ~> aliasalias ..='cd ..'alias ...='cd ../..'alias ....='cd ../../..'alias PAGER='less -r'alias Txterm='export TERM=xterm'alias XARGS='xargs -r'alias cdrecord='cdrecord -dev 0,0,0 -speed=8'alias e='vi'alias egrep='grep -E'<zkráceno>
```

Aliasy jsou užitečné například k výběru výchozí verze příkazu, který v systému existuje v několi-ka verzích, nebo k zadání výchozích voleb příkazu. Další možné použití je automatická oprava oblíbených překlepů.

U prvního slova každého jednoduchého příkazu, není-li zadáno v uvozovkách, se kontroluje, zda nejde o alias. Pokud ano, je slovo nahrazeno textem aliasu. Název aliasu a nahrazující text může obsahovat jakýkoliv platný vstup shellu včetně metaznaků s tou výjimkou, že název aliasu nemů-že obsahovat znak =. První slovo nahrazujícího textu se opět testuje, zda nejde o název aliasu, nic-méně pokud je první slovo stejné jako nahrazovaný alias, k opakované expanzi nedojde. Proto můžeme například definovat alias ls expandující se na ls -F a bash se nebude pokoušet o rekur-zivní expanzi. Je-li posledním znakem nahrazujícího textu mezera nebo tabulátor, provede se expanze aliasů i u bezprostředně následujícího slova v příkazu.

K expanzi aliasů nedochází u neinteraktivního shellu, toto chování je možné změnit nastavením volby expand_aliases prostřednictvím vestavěného příkazu shopt.

Vytváření a rušení aliasů

Aliasy se vytvářejí vestavěným příkazem alias. Má-li alias existovat trvale, definujte jej v některém z inicializačních souborů shellu. Pokud alias vytvoříte na příkazovém řádku, bude platný pouze v aktuálním shellu.

franky ~> alias *dh='df -h'*

franky ~> dh Filesystem Size Used Avail Use% Mounted on /dev/hda7 1.3G 272M 1018M 22% / /dev/hda1 121M 9.4M 105M 9% /boot /dev/hda2 13G 8.7G 3.7G 70% /home /dev/hda3 13G 5.3G 7.1G 43% /opt none 243M 0 243M 0% /dev/shm /dev/hda6 3.9G 3.2G 572M 85% /usr /dev/hda5 5.2G 4.3G 725M 86% /var franky ~> unalias *dh*

franky ~> dh bash: dh: command not found

franky ~>

Před provedením jakéhokoliv příkazu na řádku bash vždy načte nejméně jeden celý řádek vstu-pu. Aliasy se expandují při načítání příkazu, nikoliv při jeho provádění. Definujete-li tedy alias na stejném řádku jako další příkaz, nastavení aliasu nebude na daném řádku ještě platné. Příkazy následující na stejném řádku za definicí aliasu tak nebudou novým aliasem ovlivněny. Toto cho-vání se uplatňuje i při provádění funkcí. Alias se expanduje při čtení definice funkce, nikoliv při spouštění funkce, protože definice funkce je sama chápána jako složený příkaz. Důsledkem je, že aliasy definované uvnitř funkce nejsou k dispozici, dokud funkce nebude provedena. Bezpečný přístup proto je vždy aliasy definovat na samostatném řádku a nepoužívat příkaz alias ve slože-ných příkazech.
Synovské procesy nedědí nastavené aliasy. Bourne shell (sh) aliasy nezná. Více informací o funkcích naleznete v kapitole „Funkce“.

Funkce jsou rychlejší
Seznam aliasů je prohledáván až po seznamu funkcí, nalezení funkce je proto rychlejší.

I když je mechanismus aliasů snazší na pochopení, prakticky ve všech případech jsou funk

ce vhodnější metodou řešení.

Další volby bashe

Volby zobrazení

Už jsme hovořili o některých volbách bashe, vhodných pro ladicí účely. V této části budeme o vol-bách hovořit podrobněji.Seznam voleb vypíšete parametrem -o příkazu set:

willy:~> set -o

allexport	off
braceexpand	on
emacs	on
errexit	off
hashall	on
histexpand	on
history	on
ignoreeof	off
interactive-comments	on
keyword	off
monitor	on
noclobber	off
noexec	off
noglob	off
nolog	off
notify	off
nounset	off
onecmd	off

physical	off
posix	off
privileged	off
verbose	off
vi	off
xtrace	off

Popis jednotlivých voleb naleznete na informačních stránkách bashe v části Shell Built-in Com-mands -> The Set Built-in.

Většina voleb má rovněž jednoznakovou zkratku, například volba xtra-ce je ekvivalentní nastavení set -x.

Změna voleb

Nastavení voleb shellu je možné změnit buď při spouštění shellu anebo za jeho běhu. Nastavenímohou být také specifikována v konfiguračních souborech shellu.Následující příkaz spustí skript v režimu kompatibility s normou POSIX:

willy:~/scripts> bash --posix script.sh

K dočasné změně nastavení prostředí nebo k nastavení v rámci skriptu použijte příkaz set. Para-metrem -volbu zapnete, parametrem + ji vypnete:

willy:~/test> set -o *noclobber*

willy:~/test> touch test

willy:~/test> date > test bash: test: cannot overwrite existing file

willy:~/test> set +o *noclobber*

willy:~/test> date > test

Tento příklad demonstruje použití volby noclobber, která zabrání přepsání existujícího souboru operací přesměrování. Stejným způsobem se nastavují jednoznakové volby, například -u, která referenci na nenastavenou proměnnou chápe jako chybu a neinteraktivní shell v takovém přípa-dě skončí:

willy:~> echo $VAR

willy:~> set -u

willy:~> echo $VAR bash: VAR: unbound variable

Tato volba je vhodná také k detekci nekorektních přiřazovacích operací – ke stejné chybě napří-klad dojde, pokud se pokusíte přiřadit znakovou hodnotu do proměnné, která byla explicitně deklarována jako celočíselná.

Poslední příklad demonstruje použití volby noglob, která zabraňuje expanzi speciálních znaků:

willy:~/testdir> set -o *noglob*

willy:~/testdir> touch *
willy:~/testdir> ls -l *-rw-rw-r--1 willy willy 0 Feb 27 13:37 *

Shrnutí

Prostředí bashe je možné konfigurovat jak globálně, tak pro každého uživatele zvlášť. Chování

shellu ovlivňují různé konfigurační soubory.
Tyto soubory obsahují volby shellu, nastavují proměnné, definují funkce a umožňují tak různými způsoby konfigurovat chování prostředí.

Názvy proměnných je možno volit libovolně, jen nesmí dojít ke kolizi s rezervovanými proměn

nými Bourne shellu, bashe nebo se speciálními parametry.Řada znaků může mít podle kontextu dva i více významů, proto bash používá mechanismus uvo-zování, jímž se potlačuje speciální význam jednoho nebo více znaků v případech, kdy je tentovýznam nežádoucí.

K sestavení prováděného příkazu používá bash různé metody expanze textu zadaného na příka zovém řádku.

Cvičení

Pro potřeby tohoto cvičení si nastudujte manuálovou stránku příkazu useradd, protože budeme pracovat s adresářem /etc/skel, který obsahuje výchozí konfigurační soubory shellu, které se kopírují do domovských adresářů nově vytvářených uživatelů.

Nejprve několik obecných cvičení zaměřených na nastavování a zobrazování proměnných:

Vytvořte tři proměnné, VAR1, VAR2 a VAR3; nastavte jejich obsah na hodnoty „třináct“, „13“ a „šťastné a veselé“.
Vypište hodnoty všech tří proměnných.
Jde o lokální, nebo globální proměnné?
Odstraňte proměnnou VAR3.
Uvidíte zbývající dvě proměnné v novém terminálovém okně?
Upravte soubor /etc/profile tak, aby systém každého uživatele při přihlášení pozdravil (otestujte).
Pro uživatele *root* nastavte prompt na něco jako „Pozor!! Root pracuje v \w“, nejlépe v něja-kém výstražném barevném provedení.
Zajistěte, aby i nově vytvoření uživatelé měli nastaven pěkný prompt, který jim řekne, na jakém systému a v jakém adresáři pracují. Otestujte provedené změny tak, že vytvoříte nový uživatelský účet a přihlásíte se jako nový uživatel.
Napište skript, v němž do dvou proměnných přiřadíte dvě celočíselné hodnoty. Z těchto zadaných hodnot skript vypočítá plochu obdélníka. Skript by měl být komentovaný a měl by poskytovat elegantní výstup.

Nezapomeňte změnit práva skriptů příkazem chmod!

Regulární výrazy

V této kapitole budeme hovořit o následujících tématech:

Použití regulárních výrazů
Metaznaky v regulárních výrazech
Hledání vzorů v souborech nebo výstupu
Rozsahy a třídy znaků v bashi

Regulární výrazy

Co jsou to regulární výrazy?

Regulární výraz je vzor, který popisuje množinu řetězců. Regulární výrazy se vytvářejí podobně
jako aritmetické výrazy tak, že se pomocí různých operátorů kombinují menší výrazy. Základním stavebním blokem jsou regulární výrazy, jimž vyhovuje jediný znak. Většina znaků, například všechny písmena a číslice, samy o sobě představují regulární výraz, jemuž vyhovuje právě daný znak. Metaznaky se speciálním významem je možné escapovat pomocí obráceného lomítka.

Metaznaky v regulárních výrazech

Za regulárním výrazem může následovat jeden z operátorů opakování (metaznaky):

Operátor Význam

Vyhovuje mu jakýkoliv jeden znak. Předchozí výraz je nepovinný, může se vyskytovat nanejvýš jednou.

Předchozí výraz se může vyskytovat nula nebo vícekrát.

Předchozí výraz se může vyskytovat jednou nebo vícekrát.

{Předchozí výraz se může vyskytovat právě Nkrát.

{Předchozí výraz se může vyskytovat N nebo vícekrát.

{Předchozí výraz se může vyskytovat nejméně Nkrát, nejvýše Mkrát.

Není-li prvním či posledním znakem seznamu nebo posledním znakem rozsahu v seznamu,

pak definuje rozsah. Vyhovuje mu prázdný řetězec na začátku řádku, též definuje nepřítomnost znaku v rozsahu. Vyhovuje mu prázdný řetězec na konci řádku. \Vyhovuje mu prázdný řetězec na začátku či konci slova.

Operátor	Význam
\	Vyhovuje mu prázdný řetězec ne na začátku či konci slova.

\ Vyhovuje mu prázdný řetězec na začátku slova.
\ Vyhovuje mu prázdný řetězec na konci slova.

Operátory regulárních výrazů

Regulární výrazy je možné řetězit. Výslednému regulárnímu výrazu vyhovují řetězce skládající se
z podřetězců vyhovujících příslušným zřetězeným regulárním výrazům.Regulární výrazy je možné spojovat infixovým operátorem |. Výslednému regulárnímu výrazuvyhovují řetězce vyhovující kterémukoliv z takto spojených regulárních výrazů.

Opakování má vyšší prioritu před řetězením, to má větší prioritu před alternací. Části výrazů je možno uzavírat do závorek, které přepisují výchozí chování priority.

Základní a rozšířené regulární výrazy

V základních regulárních výrazech nemají znaky ?, +, {, |, (a) speciální význam, musíte je zapsat

v podobě \?, \+, \{, \|, \(a \).V dokumentaci k systému zjistíte, zda příkazy pracující s regulárními výrazy podporují základnínebo rozšířenou verzi.

Příklady s příkazem grep

Co je grep?

Příkaz grep hledá ve vstupním souboru řádky, které vyhovují zadanému vzoru. Pokud řádek najde,
vypíše jej (standardně) na výchozí výstup nebo kamkoliv jinam podle toho, jak jej spustíte. I když je grep primárně určen pro hledání v textu, délka prohledávaných řádků není omezena jinak než velikostí paměti a testovat se může shoda s libovolnými znaky na řádku. Jestliže posled-ním znakem souboru není znak nového řádku, grep jej tam automaticky doplní. Znak nového řádku nicméně slouží zároveň jako oddělovač seznamu hledaných vzorů, neexistuje tedy žádný způsob, jak tímto příkazem v souboru hledat znaky nového řádku.
Několik příkladů:

cathy ~> grep *root* /etc/passwd root:x:0:0:root:/root:/bin/bash operator:x:
11:0:operator:/root:/sbin/nologin

cathy ~> grep -n *root* /etc/passwd 1:root:x:0:0:root:/root:/bin/bash 12:operator:x:
11:0:operator:/root:/sbin/nologin

cathy ~> grep -v *bash* /etc/passwd | grep -v *nologin* sync:x:5:0:sync:/sbin:/bin/sync shutdown:x:
6:0:shutdown:/sbin:/sbin/shutdown halt:x:7:0:halt:/sbin:/sbin/halt news:x:9:13:news:/var/spool/news:
mailnull:x:47:47::/var/spool/mqueue:/dev/null xfs:x:43:43:X Font Server:/etc/X11/fs:/bin/false rpc:x:
32:32:Portmapper RPC user:/:/bin/false nscd:x:28:28:NSCD Daemon:/:/bin/false named:x:
25:25:Named:/var/named:/bin/false squid:x:23:23::/var/spool/squid:/dev/null ldap:x:55:55:LDAP
User:/var/lib/ldap:/bin/false apache:x:48:48:Apache:/var/www:/bin/false

cathy ~> grep -c *false* /etc/passwd

cathy ~> grep -i *ps* ~/.bash* | grep -v *history* /home/cathy/.bashrc:PS1="\[\033[1;44m\]$USER is in \w\[\033[0m\] "

První příkaz zobrazí ty řádky souboru /etc/passwd , které obsahují řetězec *root*.Druhý příkaz zároveň zobrazí čísla nalezených řádků.Třetím příkazem hledáme uživatele, kteří nepoužívají bash, nevypisujeme ale uživatele, kteří mají

nastavený shell nologin.Čtvrtý příkaz počítá účty, které mají jako shell nastaveno /bin/false.Poslední příkaz prohledává všechny soubory v domovském adresáři začínající na ~/.bash, hledá
se výskyt znaků *ps* bez ohledu na velikost písmen. Z výstupu se odfiltrují řádky obsahující řetě-zec *history*, takže se vyloučí případné duplicitní nálezy v souboru ~/.bash_history.Nyní se podíváme, jak můžeme v příkazu grep pracovat s regulárními výrazy.

Grep a regulární výrazy

Nepoužíváte-li Linux V následujících příkladech používáme GNU verzi příkazu grep, která podporuje rozšířené regulární výrazy. Na linuxových systémech je GNU grep výchozí instalovanou verzí. Pou-žíváte-li nějaký proprietární systém, volbou -V si ověřte, jakou verzi máte instalovánu. GNU grep můžete získat na adrese *http://gnu.org/directory*.

Hranice slov a řádků

Vyjdeme z předchozího příkladu, budeme však hledat pouze ty řádky, které slovem „root“ *začí-nají*:

```
cathy ~> grep ^root /etc/passwd root:x:0:0:root:/root:/bin/bash
```

Budeme-li se zajímat o účty, které nemají vůbec nastaven přihlašovací shell, budeme hledat řádky, které *končí* dvojtečkou:

```
cathy ~> grep :$ /etc/passwd news:x:9:13:news:/var/spool/news:
```

Chceme-li ověřit, zda se proměnná PATH exportuje ze souboru ~/.bashrc, nejprve budeme hle-dat všechny řádky s textem *export* a následně z nich vybereme řádky, na nichž začíná slovo *PATH*, abychom tak vyloučili zobrazení proměnných MANPATH a podobných:

```
cathy ~> grep export ~/.bashrc | grep '\<PATH' export
   PATH="/bin:/usr/lib/mh:/lib:/usr/bin:/usr/local/bin:/usr/ucb:/usr/
   dbin:$PATH"
```

Analogicky znamená zápis \> konec slova.Pokud chcete najít samostatné slovo (tedy řetězec obklopený mezerami), je lepší použít volbu -wtak, jak to ukazuje následující příklad, ve kterém zobrazujeme informace o kořenovém svazku:

```
cathy ~> grep -w / /etc/fstab LABEL=/ / ext3 defaults 1 1
```

Pokud byste nezadali žádnou volbu, vypsaly by se všechny řádky ze souboru.

Třídy znaků

Výraz v hranatých závorkách je tvořen seznamem znaků uzavřených mezi [a]. Vyhovuje mu kte-rýkoliv jeden znak ze zadaného seznamu. Je-li prvním znakem ^, vyhovuje mu kterýkoliv znak, který v seznamu *není* uveden. Například regulárnímu výrazu [0123456789] bude vyhovovat každá jednotlivá číslice.

V rámci hranatých závorek je možné definovat i rozsah znaků, který se zapisuje jako dva znaky oddělené pomlčkou. Takovému rozsahu vyhovuje kterýkoliv znak řazený mezi uvedené znaky včetně, používá se znaková sada a řadicí pravidla podle aktuálního lokalizačního nastavení. Například při použití lokalizačního nastavení pro jazyk C je zápis [a-d] ekvivalentní zápisu [abcd]. Ve většině lokalizačních nastavení dochází ke slovníkovému řazení, takže zápis [a-d] typicky nebu-de ekvivalentní zápisu [abcd], ale bude ekvivalentní něčemu jako [aBbCcDd]. Chcete-li dosáhnout tradiční interpretace rozsahů, nastavte proměnnou prostředí LC_ALL na hodnotu C.

A konečně, v rámci hranatých závorek je možné použít několik předdefinovaných pojmenovaných tříd znaků, například *lower* nebo *digit*. Seznam a popis předdefinovaných tříd naleznete na manu-álových nebo informačních stránkách příkazu grep.

```
cathy ~> grep [yf] /etc/groupsys:x:3:root,bin,admtty:x:5:mail:x:12:mail,postfixftp:x:50:nobody:x:99:floppy:x:19:xfs:x:43:nfsnobody:x:
65534:postfix:x:89:

cathy ~> ls *[1-9].xmlapp1.xml chap1.xml chap2.xml chap3.xml chap4.xml
```

V prvním příkladu zobrazujeme všechny řádky obsahující znaky *y* nebo *f*, ve druhém příkladu používáme rozsah znaků přímo v příkazu ls.

Zástupné znaky

Tečce vyhovuje libovolný znak. Pokud chcete najít všechna pětiznaková anglická slova začínající na *c* a končící na *h* (výborné při luštění křížovek), můžete to udělat takto:

```
cathy ~> grep '\<c...h\>' /usr/share/dict/words catch clash cloth coach couch cough crash crush
```

Pokud byste chtěli hledat řádky obsahující přímo tečku, spusťte příkaz grep s volbou -F. Hvězdička znamená libovolný počet opakování předchozího výrazu. Následující příklad hledá všechna slova začínající na *c* a končící na *h* (mezi nimi je libovolný počet opakování libovolného znaku, tj. *):

```
cathy ~> grep '\<c.*h\>' /usr/share/dict/words caliph cash catch cheesecloth cheetah ...
```

Pokud byste chtěli hledat řádky obsahující přímo hvězdičku, spusťte příkaz grep s volbou -F:

```
cathy ~> grep * /etc/profile

cathy ~> grep -F '*' /etc/profile for i in /etc/profile.d/*.sh ; do
```

Hledání podle vzoru v bashi

Rozsahy znaků

Kromě příkazu grep a jím používaných regulárních výrazů můžete hledání podle vzoru používat
i přímo v shellu, bez nutnosti použití externího programu.Jak už víme, hvězdička (*) a otazník (?) zastupují libovolný řetězec, respektive libovolný znak.Zapíšete-li tyto znaky v uvozovkách, ponechávají si svůj literální význam:

```
cathy ~> touch "*"

cathy ~> ls "*" *
```

Pomocí hranatých závorek je také možné definovat seznam znaků nebo rozsah znaků, pokud dvojici znaků oddělíte pomlčkou. Například:

```
cathy ~> ls -ld [a-cx-z]*drwxr-xr-x 2 cathy cathy 4096 Jul 20 2002 app-defaults/drwxrwxr-x 4 cathy cathy 4096 May 25 2002 arabic/drwxrwxr-x 2 cathy cathy 4096 Mar 4 18:30 bin/drwxr-xr-x 7 cathy cathy 4096 Sep 2 2001 crossover/drwxrwxr-x 3 cathy cathy 4096 Mar 22 2002 xml/
```

Tímto příkazem vypisujeme všechny soubory, jejichž názvy začínají na *a*, *b*, *c*, *x*, *y* nebo *z*. Pokud je prvním znakem v hranatých závorkách *!* nebo ^, vyhovují zápisu všechny neuvedené znaky. Pokud chcete ve výčtu znaků uvést pomlčku, uveďte ji jako první nebo jako poslední znak výčtu. Řazení a vyhledávání závisí na lokalizačním nastavení a také na hodnotě proměnné LC_COLLATE, je-li nastavena. Je-li nastaveno slovníkové řazení, může být zápis [a-cx-z] interpreto-ván například jako [aBbCcXxYyZz]. Chcete-li zajistit tradiční interpretaci intervalů, nastavte pro-měnnou LC_COLLATE nebo LC_ALL na „C“.

Třídy znaků

Třídy znaků je možné definovat v hranatých závorkách zápisem [:třída:], kde *třída* je definována normou POSIX a může mít některou z následujících hodnot: *alnum*, *alpha*, *ascii*, *blank*, *cntrl*, *digit*, *graph*, *lower*, *print*, *punct*, *space*, *upper*, *word* nebo *xdigit*.

Například:

```
cathy ~> ls -ld [[:digit:]]*drwxrwxr-x 2 cathy cathy 4096 Apr 20 13:45 2/

cathy ~> ls -ld [[:upper:]]*drwxrwxr--3 cathy cathy 4096 Sep 30 2001 Nautilus/drwxrwxr-x 4 cathy cathy 4096 Jul 11 2002 OpenOffice.org1.0/-rw-rw-r--1 cathy cathy 997376 Apr 18 15:39 Schedule.sdc
```

Je-li zapnuta volba extglob shellu (pomocí vestavěného příkazu shopt), zpracovávají se i někte-ré další operátory pro složitější vyhledávání výrazů. Více informací naleznete na informačních stránkách bashe, v části Basic shell features -> Shell Expansions -> Filename Expansion -> Pattern Matching.

Shrnutí

Regulární výrazy představují silný nástroj pro výběr konkrétních řádků ze souborů nebo z výstu-pu. Regulární výrazy používá celá řada unixových příkazů: vim, perl, databáze PostgreSQL a další. Pomocí externích knihoven jsou přístupné ve většině jazyků a aplikací a naleznete je i na neunixových systémech. S regulárními výrazy dokáže pracovat například i Excel z balíku Micro-soft Office. V této kapitole jsme pracovali převážně s příkazem grep, který je neoddělitelnou sou-částí každého unixového prostředí.

> Příkaz grep toho dokáže mnohem více než jen těch několik úkonů, které jsme si zde předvedli – používali jsme jej pouze jako příklad pro práci s regulárními výrazy. Ke GNU verzi příkazu grep existuje rozsáhlá dokumentace, kterou vám rozhodně doporučujeme přečíst!

Cvičení

Následující cvičení vám pomohou zažít si práci s regulárními výrazy.

Zobrazte seznam uživatelů systému, kteří jako výchozí přihlašovací shell používají bash.
Zobrazte všechny řádky souboru /etc/group, které začínají řetězcem „daemon“.
Vypište z téhož souboru všechny řádky, které daný řetězec neobsahují.
Vypište informace o *localhostu* ze souboru /etc/hosts, zobrazte čísla řádků, na nichž se hledaný řetězec vyskytuje, a spočítejte počet výskytů.
Zobrazte seznam podadresářů /usr/share/doc, které obsahují informace o shellech.
Kolik tyto podadresáře obsahují souborů README? Nezapočítávejte soubory ve tvaru *README.něco*.
Pomocí příkazu grep vypište názvy těch souborů z vašeho domovského adresáře, které byly změněny v posledních deseti hodinách. Nevypisujte adresáře.

Vytvořte skript, který bude v přehledném tvaru vypisovat soubory z předchozího bodu zadání.
Jak byste pomocí příkazu grep nahradili příkaz wc -l?
Ze souboru /etc/fstab vypište lokální disková zařízení.
Vytvořte skript, který v souboru /etc/passwd ověří existenci zadaného uživatele. Pro tuto chvíli zadejte hledaného uživatele přímo ve skriptu, nezabývejte se předáváním paramet-rů.
Zobrazte takové konfigurační soubory z adresáře /etc, které ve svém názvu obsahují čísla.

Editor sed

Na konci této kapitoly budete vědět více o následujících tématech:

Co je to sed
Interaktivní použití editoru sed
Regulární výrazy a neinteraktivní editace
Použití příkazu sed ve skriptech

> Toto je jen úvod
>
> Následující text rozhodně není úplný popis a není zamýšlen jako uživatelský manuál k sedu. Tuto kapitolu zařazujeme pouze jako základ pro některá zajímavá témata z násle-dujících kapitol a také proto, že každý zkušený uživatel by měl mít alespoň základní zna-losti o editoru sed a jeho možnostech. Podrobnější informace získáte na informačních a manuálových stránkách editoru.

Úvod

Co je to sed?

Editor sed (Stream EDitor) je určen k základním transformacím textu načítaného ze souboru nebo roury. Výsledek je posílán na standardní výstup. Syntaxe příkazu sed neumožňuje zadat výstupní soubor, výsledek však lze uložit do souboru prostřednictvím přesměrování. Editor nemodifikuje původní vstupní soubor.

Rozdíl mezi sedem a jinými editory, jako jsou vi nebo ed, spočívá v jeho schopnosti zpracovávat text předávaný rourou. S editorem za jeho běhu nijak nekomunikujete, proto se také někdy ozna-čuje jako *dávkový editor*. Díky tomu je vhodný k editačním účelům ve skriptech a výrazně usnad-ňuje opakované editace textových souborů. Jakmile potřebujete změnit nějaký text ve větším počtu souborů, je sed tím pravým nástrojem.

Příkazy editoru sed

Program sed dokáže nahrazovat a mazat text podle vzoru definovaného pomocí regulárních výra-zů podobných těm, které používá příkaz grep, viz kapitolu „Příklady s příkazem grep“.
Příkazy pro editaci jsou podobné těm, které používá editor vi:

Příkaz Význam

Přidá text za aktuální řádek.

Nahradí text na aktuálním řádku jiným řádkem.

Vymaže text.

Vloží text před aktuální řádek.

Vytiskne text.

Čte soubor.

Hledá a nahrazuje text.

Zapisuje do souboru.

Příkazy editoru sed

Kromě editačních příkazů můžete sedu také zadat parametry. Jejich přehled uvádí následující tabulka:

Volba	Význam
	Přidá skript k příkazům, které budou prováděny při zpracování vstupu.
	Přidá příkazy v souboru k příkazům, které budou prováděny při zpracování vstupu.
	Tichý režim.
	Vypíše verzi programu a skončí.

Volby příkazu sed

Další informace naleznete na informačních stránkách programu, zde uvádíme jen nejčastější příkazy a volby.

Interaktivní editace

Vytištění řádků obsahujících vzor

Jde pouze o počáteční příklad, něco takového můžete samozřejmě udělat i příkazem grep. Ope-raci „najdi a nahraď“ už byste ale s jeho pomocí neudělali.Takto vypadá náš vstupní soubor:

```
[jura@jv2 sed]$ cat -n priklad
     1 Toto je první řádek příkladu.
     2 Obsahuje chiby.
     3 Spoustu chib.
     4 Obsahuje tolik chib, že je mi z těch chib špatně.
     5 Tento řádek neobsahuje žádné chyby.
     6 A toto je poslední řádek.
```

Příkazem sed najdeme všechny řádky, které obsahují hledaný vzor, v tomto případě „chib“. Nale-zené řádky vypisujeme příkazem p:

```
[jura@jv2 sed]$ sed '/chib/p' priklad Toto je první řádek příkladu. Obsahuje chiby.
Obsahuje chiby.Spoustu chib.Spoustu chib.Obsahuje tolik chib, že je mi z těch chib špatně.Obsahuje tolik chib, že je mi z těch chib špatně.Tento řádek neobsahuje žádné chyby.A toto je poslední řádek.
```

Můžete si všimnout, že sed vypsal celý soubor, řádky obsahující hledaný text jsou vypsány dva-krát. Takové chování ovšem nepotřebujeme. Chceme-li vytisknout pouze ty řádky, které odpoví-dají hledanému vzoru, použijeme volbu -n:

```
[jura@jv2 sed]$ sed -n '/chib/p' priklad
Obsahuje chiby.
Spoustu chib.
Obsahuje tolik chib, že je mi z těch chib špatně.
```

Smazání řádků obsahujících vzor

Používáme stejný vstupní soubor. Nyní budeme chtít zobrazit pouze ty řádky, které *neobsahují* zadaný řetězec:

```
[jura@jv2 sed]$ sed '/chib/d' priklad
Toto je první řádek příkladu.
Tento řádek neobsahuje žádné chyby.
A toto je poslední řádek.
```

Příkaz d maže řádky odpovídající vzoru a ve výsledku se tak vypíší jen řádky, které vzor neobsa-hují.Nalezení řádku, který začíná určitým vzorem a končí jiným vzorem, vypadá takto:

```
[jura@jv2 sed]$ sed -n '/^Obsahuje.*špatně.$/p' priklad Obsahuje tolik chib, že je mi z těch chib špatně.
```

Rozsahy řádků

Nyní budeme chtít odstranit řádky s chybami. V našem případě to jsou řádky 2 až 4. Můžeme zadat požadovaný rozsah řádků a příkaz d:

```
[jura@jv2 sed]$ sed '2,4d' priklad Toto je první řádek příkladu. Tento řádek
neobsahuje žádné chyby. A toto je poslední řádek.
```

Smazání řádků od zadaného po poslední můžeme provést takto:

```
[jura@jv2 sed]$ sed '3,$d' priklad Toto je první řádek příkladu. Obsahuje
chiby.
```

Takto vypíšeme pouze první dva řádky příkladu.
Následující příklad vytiskne řádky počínaje řádkem, který obsahuje vzor „chiby“ a konče řádkem obsahujícím vzor „řádek“:

```
[jura@jv2 sed]$ sed -n '/chiby/,/řádek/p' priklad Obsahuje chiby.
Spoustu chib.
Obsahuje tolik chib, že je mi z těch chib špatně.
Tento řádek neobsahuje žádné chyby.
```

Hledání a nahrazování

Nyní už nebudeme jen vybírat nebo potlačovat určité řádky, ale rovnou provedeme nalezení a opravu chyb:

```
[jura@jv2 sed]$ sed 's/chib/chyb/' prikladToto je první řádek příkladu.Obsahuje chyby.Spoustu chyb.Obsahuje tolik chyb, že je mi z těch chib špatně.Tento řádek neobsahuje žádné chyby.A toto je poslední řádek.
```

Všimněte si, že výsledek neodpovídá našim představám – na čtvrtém řádku byl nahrazen pouze první výskyt hledaného řetězce, druhá „chiba“ tam zůstala.

Příkazem g řekneme sedu, že má zpracovávat celý řádek, že se nemá zastavit na prvním výskytu hledaného řetězce:

```
[jura@jv2 sed]$ sed 's/chib/chyb/g' prikladToto je první řádek příkladu.Obsahuje chyby.Spoustu chyb.Obsahuje tolik chyb, že je mi z těch chyb špatně.Tento řádek neobsahuje žádné chyby.A toto je poslední řádek.
```

Následující příklad ukazuje vložení textu „>“ na začátek každého řádku:

```
[jura@jv2 sed]$ sed 's/^/> /' priklad> Toto je první řádek příkladu.> Obsahuje chiby.> Spoustu chib.> Obsahuje tolik chib, že je mi z těch chib špatně.> Tento řádek neobsahuje žádné chyby.> A toto je poslední řádek.
```

A vložení textu na konec řádku:

```
[jura@jv2 sed]$ sed 's/$/EOL/' priklad Toto je první řádek příkladu.EOL Obsahuje chiby.EOL Spoustu chib.EOL Obsahuje tolik chib, že je mi z těch chib špatně.EOL Tento řádek neobsahuje žádné chyby.EOL A toto je poslední řádek.EOL
```

Více příkazů pro nalezení a náhradu lze zadat pomocí volby -e:

```
[jura@jv2 sed]$ sed -e 's/chib/chyb/g' -e 's/poslední/závěrečný/g' priklad
Toto je první řádek příkladu.
Obsahuje chyby.
Spoustu chyb.
Obsahuje tolik chyb, že je mi z těch chyb špatně.

Tento řádek neobsahuje žádné chyby. A toto je závěrečný řádek.
```

Nezapomínejte, že standardně sed zapisuje výsledek na standardní výstup, tedy nejspíše do ter-minálového okna. Pokud budete chtít výsledek uložit do souboru, použijte přesměrování:

sed volby ‘nějaký_výraz’ vstupní_soubor > výstupní_soubor

> Více příkladůŘadu příkladů použití editoru sed naleznete ve spouštěcích skriptech svého počítače,typicky v adresářích /etc/init.d nebo /etc/rc.d/init.d. Přejděte do adresáře s inici-alizačními skripy a zadejte následující příkaz:
>
> grep *sed* *

Neinteraktivní editace

Načtení příkazů sedu ze souboru

Více příkazů sedu můžete zapsat do jednoho souboru a provést je pomocí volby -f. Při vytváření souboru zajistěte:

Na konci souboru nesmí být nadbytečné mezery, konce řádků a podobné znaky.
V souboru nesmí být používány uvozovky.
Při zadávání textu k přidání či náhradě ukončujte všechny řádky kromě posledního obráceným lomítkem.

Uložení výstupního souboru

Uložení výstupu se zajišťuje prostřednictvím operátorů přesměrování. Následující příklad ukazuje jednoduchý skript, který z čistého textového souboru vytvoří HTML stránku:

```
[jura@jv2 sed]$ cat script.sed1i\<html>\<head><title>sed-em generované html</title></head>\<body bgcolor="#ffffff">\<pre>
$a\</pre>\</body>\</html>[jura@jv2 sed]$ cat txt2html.sh#!/bin/bash

# Tento jednoduchy skript konvertuje text na HTML.
# Nejprve odstranime vsechny znaky noveho radku, aby k pripojeni
# doslo jen jednou, pak nove radky obnovime


echo "konvertuji $1..."


SCRIPT="./script.sed"
NAME="$1"

TEMPFILE="/var/tmp/sed.$PID.tmp" sed "s/\n/^M/" $1 | sed -f $SCRIPT | sed "s/^M/\n/" > $TEMPFILE mv $TEMPFILE $NAME

echo "hotovo."
```

Proměnná $1 obsahuje první parametr předaný skriptu, v našem případě to bude název konver-tovaného souboru:

```
[jura@jv2 sed]$ cat test první řádek druhý řádek třetí řádek
```

Více informací o pozičních parametrech se dozvíme v kapitole „Podmíněné příkazy“.

```
[jura@jv2 sed]$ ./txt2html.sh testkonvertuji test...hotovo.[jura@jv2 sed]$ cat test<html><head><title>sed-em generované
html</title></head><body bgcolor="#ffffff"><pre>první řádekdruhý řádektřetí řádek</pre></body></html>
```

Toto řešení samozřejmě není dokonalé, snažíme se pouze demonstrovat možnosti sedu. V kapi-tole „Proměnné awku“ naleznete lepší řešení daného problému, s použitím konstrukcí *BEGIN* a *END* programu awk.

> Jednodušší sed
>
> Kvalitní textové editory s podporou zvýrazňování syntaxe typicky rozeznávají i syntaxi pří-kazu sed. Tato pomoc může být užitečná, zejména pokud se vám daří zapomínat na zpět-ná lomítka a podobně.

Shrnutí

Textový editor sed je mocný řádkový nástroj, kterým můžete zpracovávat datové proudy – doká-že načítat vstup z roury. Je tak velmi vhodný k neinteraktivnímu použití. Editor sed používá syn-taxi podobnou editoru vi a dokáže zpracovávat regulární výrazy.

Příkazy může sed načíst z příkazového řádku nebo ze skriptu. Velmi často se používá k nalezení a náhradě opakujícího se textu.

Cvičení

Následující cvičení by vám měla demonstrovat, co všechno sed dokáže.
Vytvořte seznam souborů v adresáři scripts, jejichž název končí znaky „.sh“. Uložte seznam do pomocného souboru.
Vytvořte seznam souborů v adresáři /usr/bin, jejichž druhé písmeno názvu je „a“. Ulož-te seznam do pomocného souboru.
Smažte první tři řádky v obou pomocných souborech.
Vypište na standardní výstup jen ty řádky, které obsahují znaky "an".
Vytvořte soubor s příkazy sedu, které provedou předchozí dva úkoly. Přidejte do soubo-ru další příkaz, který před každý řádek obsahující řetězec „man“ přidá řádek s textem „*** toto má asi něco společného s manuálovými stránkami ***“. Ověřte výsledek.
Jako vstup použijte podrobný výpis obsahu kořenového adresáře. Vytvořte soubor s pří-kazy sedu, který bude vyhledávat symbolické odkazy a klasického soubory. Je-li položka výpisu symbolickým odkazem, přidejte před ni řádek s textem „-- toto je

symbolický odkaz --“. Je-li položka klasickým souborem, přidejte na stejný řádek komentář „<--- toto je sou-bor“.
Vytvořte skript, který v zadaném souboru zobrazí řádky končící mezerou. Skript by měl samozřejmě používat sed a vypisovat uživateli rozumné informace.

Programovací jazyk GNU awk

V této kapitole budeme hovořit o následujících tématech:

Co je to gawk
Použití příkazů gawk na příkazovém řádku
Jak pomocí gawk formátovat text
Jak gawk používá regulární výrazy
Gawk ve skriptech
Gawk a proměnné

> Aby to bylo zajímavější Stejně jako v případě sedu byly napsány celé knihy o různých verzích awku. Tento úvod-ní text rozhodně není úplný a jeho cílem je pouze napomoci vám v pochopení příkladů v dalších kapitolách. Jako další zdroj informací vám doporučujeme přímo dokumentaci ke GNU awku: „GAWK: Effective AWK Programming: A User's Guide for GNU Awk“. V češ-tině vyšla podrobná publikace Awk & sed.

Začínáme s gawkem

Co je to gawk?

Gawk je GNU verze známého unixového programu awk, což je další oblíbený editor textových streamů. Program awk je velmi často jen odkazem na gawk, v dalším textu proto budeme pou-žívat označení awk.

Základní funkcí awku je hledat v souboru řádky nebo jiné textové jednotky obsahující jeden nebo více požadovaných vzorů. Jestliže řádek vyhovuje vzoru, provede se s ním požadovaná operace. Programy v awku se značně liší od programů ve většině jiných jazyků, protože jde o „daty ovlá-dané“ programy: Popíšete data, s nimiž chcete pracovat, a co s nimi chcete udělat. Většina ostat-ních jazyků je „procedurálních“ – podrobně popisujete jednotlivé kroky, které chcete provádět. Při použití procedurálních jazyků je popis zpracovávaných dat obvykle mnohem komplikovanější. Programy v awku se proto typicky snadno vytvářejí i čtou.

> Co znamená název
>
> Jazyk AWK vytvořila v 70. letech trojice programátorů – Aho, Kernighan a Weinberger. Název jazyka je tvořen prvními písmeny jejich jmen. Stejně dobře se tedy jazyk mohl jmenovat třeba „WAK“.

Příkazy awku

Při spouštění awku mu zadáváte *program*, který příkazu říká, co má dělat. Program se skládá z posloupnosti *pravidel*. (Může obsahovat i definice funkcí, smyčky, podmínky a další programo-vé konstrukce, což jsou pokročilejší funkce, kterými se nyní nebudeme zabývat.) Každé pravidlo definuje jeden hledaný vzor a jednu akci prováděnou nad nalezeným vzorem.

Je několik možností, jak awk spouštět. Pokud je program krátký, nejjednodušší metoda je přímé spuštění z příkazového řádku:

awk PROGRAM vstupní_soubor(y)

Provádíte-li složitější změny, případně pravidelně a nad mnoha soubory, je pohodlnější zadat pří-kazy pro awk ve skriptu. Pak vypadá spuštění takto:

awk -f SOUBOR_S_PROGRAMEM vstupní_soubor(y)

Příkazy pro tisk

Tisk vybraných položek

Příkaz print vypíše ze vstupního souboru vybraná data.Jakmile awk přečte ze vstupního souboru řádek, rozdělí jej na jednotlivé položky na základě
vstupního separátoru, FS, což je proměnná programu awk (viz kapitolu „Výstupní oddělovače“).Výchozí hodnota této proměnné je jedna nebo více mezer či tabulátorů.Tímto rozdělením dojde k naplnění proměnných $1, $2, $3, ..., $N, které obsahují první, druhou,
třetí až poslední položku vstupního řádku. Proměnná $0 (nula) obsahuje celý vstupní řádek. Zná-zorňuje to následující obrázek, na kterém vidíme výstup příkazu df v šesti sloupcích:

Výstup příkazu ls -l má devět sloupců. Příkaz print s těmito poli pracuje následujícím způsobem:

```
[jura@jura tldp]$ ls -l | awk '{print $5 $9}' 27627bash.sxw 5995091intro.ps 216885intro.sxw 6056354intro2.ps
```

Tímto příkazem jsme vytiskli pátý sloupec, obsahující velikost souboru, a poslední, devátý slou-pec, název souboru. Výstup není příliš dobře čitelný, protože sloupce od sebe nejsou odděleny. Pokud byste v příkazu print oddělili názvy tisknutých polí čárkou, bude se mezi nimi vypisovat standardní oddělovač, kterým je obvykle mezera.

Formátování výpisu

Pokud se nespokojíte s formátováním standardním oddělovačem, můžete si formátování zajistit vlastními silami:

```
[jura@jura tldp]$ ls -lhd * | grep -v celkem | \ awk '{print "Velikost souboru " $9 " je " $5 "B"}'
Velikost souboru bash.sxw je 27KB Velikost souboru intro.ps je 5,8MB
Velikost souboru intro.sxw je 212KB Velikost souboru intro2.ps je 5,8MB
```

Všimněte si zpětného lomítka, které umožňuje pokračování dlouhého vstupu na dalším řádku, aniž by shell začal první zadaný řádek interpretovat. Vstup zadávaný na příkazovém řádku sice není délkově omezen, velikost monitoru a šířka papíru však omezeny jsou. Díky zpětnému lomítku můžete výše uvedený zápis rovnou překopírovat do terminálového okna.

Volbou -h příkazu ls zapínáme lidsky formátovaný výpis velikosti větších souborů. Je-li vypiso-ván obsah adresáře, vypíše příkaz ls i celkovou velikost bloků v adresáři. Ta nás ovšem nezajímá, proto explicitně uvádíme hvězdičku a vypisujeme tak pouze všechny soubory v adresáři. Ze stej-ného důvodu přidáváme i volbu -d pro případ, že by se hvězdička expandovala na adresář. Již zmíněné zpětné lomítko signalizuje pokračování řádku, viz kapitolu „Escapování znaků“. Vytisknout můžete libovolný počet sloupců v libovolném pořadí. Následující příklad vypisuje tři nejvíce zaplněné diskové oddíly:

```
[jura@jura ~]$ df -h | sort -rnk 5 | head -3 | awk '{ print "Svazek " $6 "\t: Zaplněn z " $5 "!" }'
Svazek /home/jura/fs : Zaplněn z 38%!Svazek /home/jura/storage : Zaplněn z 38%!Svazek /home/jura/fs-public : Zaplněn z 38%!
```

Následující tabulka uvádí přehled speciálních formátovacích znaků:

Sekvence Význam

\Zvonek \Nový řádek \Tabulátor

Formátovací znaky awku

Uvozovky, symbol dolaru a další metaznaky je nutné escapovat zpětným lomítkem.

Příkaz print a regulární výrazy

Pomocí lomítek je možné definovat regulární výraz. Vstup bude porovnáván s tímto regulárním výrazem a dále se zpracují jen ty řádky, které výrazu vyhovují. Celá syntaxe je následující:

```
awk '/výraz/ { program }'
```

Následující příklad vypisuje obsazení jen lokálních diskových oddílů, síťové disky nebudou uvedeny:

```
kelly is in ~> df -h | awk '/dev\/hd/ { print $6 "\t: " $5 }'/ : 46%/boot : 10%/opt : 84%/usr : 97%/var : 73%/.vol1 : 8%

kelly is in ~>
```

Lomítka mají speciální význam, proto je nutné je v regulárním výrazu escapovat.V dalším příkladu hledáme v adresáři /etc soubory končící na „.conf“ a začínající znaky „a“ nebo„x“. Používáme rozšířenou syntaxi regulárních výrazů:

```
[jura@jura etc]$ ls -l | awk '/\<(a|x).*\.conf$/ { print $9 }' auditd.conf xinetd.conf
```

Příklad ukazuje speciální význam tečky v regulárních výrazech: První tečka znamená, že nás zajímají jakékoliv znaky za prvním hledaným řetězcem, druhá tečka je escapována, protože je součástí hledaného řetězce.

Speciální texty

Chcete-li jednorázově vypsat text před zpracovávanými daty, použijte příkaz BEGIN:

```
[jura@jura etc]$ ls -l | \awk 'BEGIN {print "Nalezené soubory:"} /\<(a|x).*\.conf$/ { print $9 }'
Nalezené soubory: auditd.conf xinetd.conf
```

Analogicky lze příkazem END vypsat text po zpracování celého vstupu:

```
[jura@jura etc]$ ls -l | \awk 'END {print "Ještě nějaké přání?"} /\<(a|x).*\.conf$/ { print $9 }'
auditd.conf xinetd.conf Ještě nějaké přání?
```

Skripty v awk

Jakmile začnete vytvářet delší příkazy, budete je možná chtít ukládat do skriptů, abyste je mohli používat opakovaně. Skripty pro awk obsahují jednotlivé příkazy definující vzory a akce. Jako příklad uvádíme skript, který nás upozorní na zaplněný diskový oddíl:

```
kelly is in ~> cat diskrep.awkBEGIN { print "*** POZOR POZOR POZOR ***" }/\<[8|9][0-9]%/ { print "Oddíl " $6 "\t: zaplněn z " $5
"!" }END { print "*** Chceme peníze na nový disk! ***" }

kelly is in ~> df -h | awk -f diskrep.awk
*** POZOR POZOR POZOR ***
Oddíl /usr : zaplněn z 97%!
*** Chceme peníze na nový disk! ***


kelly is in ~>
```

Nejprve awk vytiskne úvodní text a pak zpracuje řádky, které obsahují slovo začínající osmičkou nebo devítkou, následované číslicí a znakem procenta. Nakonec se vypíše závěrečný text.

Zvýrazňování syntaxe

Awk je programovací jazyk, jehož syntaxi rozeznává většina editorů, které dokážou zvýrazňovat syntaxi „běžných“ jazyků, jako je C, bash, HTML a podobně.

Proměnné awku

Při zpracovávání vstupního souboru používá awk několik proměnných. Některé je možné edito-vat, jiné jsou jen pro čtení.

Oddělovač vstupních polí

Oddělovač polí je buď jeden znak nebo regulární výraz, který řídí, jak awk rozděluje načítanéřádky na jednotlivá pole. Ve vstupním řádku se hledají posloupnosti znaků odpovídající defino-vanému oddělovači, jednotlivá pole jsou tvořena textem mezi těmito sekvencemi oddělovačů.

Oddělovač je uložen ve vestavěné proměnné FS. Pozor, jde o něco trochu jiného, než je pro-měnná IFS, používaná shelly kompatibilními s normou POSIX.Hodnotu oddělovače polí je možné změnit pomocí operátoru přiřazení (=). Vhodná chvíle jetypicky hned na začátku programu, ještě před zahájením zpracování vstupu, aby byl už první načí-taný řádek zpracován se správnými oddělovači. Poslouží k tomu již známý příkaz BEGIN.

Následující příklad ukazuje, jak vypsat jména a popis uživatelů definovaných v systému:

```
kelly is in ~> awk 'BEGIN { FS=":" } { print $1 "\t" $5 }' /etc/passwd --output omitted--

kelly       Kelly Smith
franky      Franky B.
eddy        Eddy White
willy       William Black
cathy       Catherine the Great
sandy       Sandy Li Wong
```

Ve skriptu by to vypadalo takto:

kelly is in ~> cat printnames.awkBEGIN { FS=":" }{ print $1 "\t" $5 }

kelly is in ~> awk -f printnames.awk /etc/passwd --vynecháno--
Správná volba oddělovače je velmi důležitá, jinak mohou vznikat problémy. Představte si například vstup v následujícím formátu:

Sandy L. Wong, 64 Zoo St., Antwerp, 2000X

Vytvořite příkaz nebo skript, který bude vypisovat jednotlivé záznamy:

awk 'BEGIN { FS="," } { print $1, $2, $3 }' vstupní_soubor

Jenomže o několik řádků níž se v souboru objeví člověk s akademickým titulem:

Sandy L. Wong, PhD, 64 Zoo St., Antwerp, 2000X

Tento řádek bude pochopitelně zpracován špatně. V případě potřeby budete muset pomocí dal-ších příkazů awk nebo sed unifikovat formát vstupního souboru.Výchozí nastavení oddělovače je jedna nebo více mezer nebo tabulátorů.

Výstupní oddělovače Oddělovač výstupních polí

Za normálních okolností jsou jednotlivá pole na výstupu oddělována mezerami. Bude to zjevné v okamžiku, kdy použijete správnou syntaxi příkazu print, kdy jednotlivé položky oddělujete čár-kami:

kelly@octarine ~/test> cat test záznam1 data1 záznam2 data2

kelly@octarine ~/test> awk *'{ print $1 $2}'* test záznam1data1 záznam2data2

kelly@octarine ~/test> awk *'{ print $1, $2}'* test záznam1 data1 záznam2 data2

Pokud čárky neuvedete, bude příkaz print chápat vypisované údaje jako jedinou položku,a nebude proto používat výstupní oddělovač, definovaný proměnnou OFS.Jako oddělovač výstupních polí můžete pomocí této proměnné nastavit libovolný řetězec.

Oddělovač výstupních záznamů

Výstupu celého příkazu print říkáme *výstupní záznam*. Výsledkem každého příkazu print je jeden výstupní záznam, za nímž následuje řetězec zvaný *oddělovač výstupních záznamů*, ORS. Výchozí hodnota této proměnné je „\n“, tedy oddělovač řádku, a každý příkaz print tak vygene-ruje samostatný výstupní řádek.

Organizaci výstupních polí a záznamů můžete změnit prostřednictvím nastavení proměnných OFS a ORS:

kelly@octarine ~/test> awk *'BEGIN { OFS=";" ; ORS="\n--> \n" } \ {print $1,$2}'* test
záznam1;data1 --> záznam2;data2 -->
Pokud proměnná ORS nebude obsahovat oddělovač řádku, bude celý výstup příkazu zapsán na jediný řádek.

Počet záznamů

Vestavěná proměnná NR obsahuje počet zpracovávaných záznamů. Inkrementuje se po načtení každého řádku. Můžete ji použít na konci zpracování ke zjištění celkového počtu záznamů anebo průběžně na jednotlivých řádcích:

kelly@octarine ~/test> cat processed.awk BEGIN { OFS="-" ; ORS="\n--> hotovo\n" } { print "Číslo záznamu " NR ":\t" $1,$2 } END { print "Celkem zpracováno záznamů: " NR }

kelly@octarine ~/test> awk -f processed.awk test Číslo záznamu 1: záznam1-data1 --> hotovo Číslo záznamu 2: záznam2-data2 --> hotovo Celkem zpracováno záznamů: 2 --> hotovo

Uživatelem definované proměnné

Kromě vestavěných proměnných si můžete definovat i vlastní proměnné. Jakmile awk narazí na odkaz na proměnnou, která neexistuje (nebyla dříve definována), dojde k vytvoření této proměn-né a jejímu nastavení na prázdný řetězec. Při dalších odkazech bude proměnná obsahovat hod-notu, která do ní byla uložena naposledy. Hodnota proměnné může být textová nebo číselná. Pro-měnným lze přiřazovat i hodnoty vstupních polí.

Hodnoty lze přiřazovat buď přímo operátorem = nebo lze použít aktuální hodnotu proměnné v kombinaci s dalšími operátory:

kelly@octarine ~> cat trzby

20021009	20021013	konzultace	BigComp	2500
20021015	20021020	skoleni	EduComp	2000
20021112	20021123	vyvoj	SmartComp	10000
20021204	20021215	skoleni	EduComp	5000

kelly@octarine ~> cat total.awk{total=total + $5 }{ print "Poslat " $4 " fakturu na " $5 " Kč"}END { print "----------\nTržby celkem: " total }

kelly@octarine ~> awk -f total.awk testPoslat BigComp fakturu na 2500 KčPoslat EduComp fakturu na 2000 KčPoslat SmartComp fakturu na 10000 KčPoslat EduComp fakturu na 5000 Kč

Tržby celkem: 19500

Je možné použít i zkrácené zápisy ve stylu jazyka C jako proměnná+=hodnota.

Další příklady

Když použijeme awk skript, bude příklad z kapitoly „Uložení výstupního souboru“ mnohem snazší:

```
kelly@octarine ~/html> cat text2html.awkBEGIN { print "<html>\n<head><title>Awkem generované
HTML</title></head>\n<bodybgcolor=\"#ffffff\">\n<pre>" }{print $0 }END { print "</pre>\n</body>\n</html>" }
```

I použití skriptu je mnohem jednodušší než v případě sedu:

```
kelly@octarine ~/html> awk -f text2html.awk testfile > file.html
```

Příklady použití awk v systému

Opět vás odkazujeme na adresář s inicializačními skripty systému. Mnohem praktičtější pří klady častého použití příkazu awk naleznete například následujícím příkazem:

grep *awk* /etc/init.d/*

Příkaz printf

Větší kontrolu nad výstupním formátem získáte, pokud místo příkazu print použijete příkaz printf. Pomocí tohoto příkazu je možné pro jednotlivé položky definovat jejich šířku a zadat celou řadu formátovacích parametrů číselných údajů (například použitá číselná soustava, zda vypisovat exponent, znaménko, kolik vypisovat číslic za desetinnou čárkou a podobně). Dosáhnete toho pomocí *formátovacího řetězce*, který definuje, jak a kde vypisovat jednotlivé údaje.

Syntaxe je stejná jako u příkazu printf v jazyce C. Podrobný popis naleznete v každé úvodní učebnici jazyka C a samozřejmě také na informačních stránkách programu awk.

Shrnutí

Nástroj awk je interpretem speciálního programovacího jazyka, jehož prostřednictvím lze pomocíněkolika řádků kódu provádět jednoduchá přeformátování textu. Jeho GNU verzi naleznete vesvém systému pod názvem gawk.

Program čte řádky vstupního souboru a rozeznává jednotlivé vstupní sloupce. Nejčastější formou

pro filtraci a formátování definovaných údajů je příkaz print.Za běhu je možné jednoduše definovat proměnné a snadno počítat součty, statistiky a další údajenad vstupními daty. Proměnné a příkazy je možné definovat i ve skriptech.
Další užitečné informace o programu awk:

Jazyk awk je na Unixech a podobných systémech stále velmi rozšířen, pro stejné účely se ale dnes častěji používá jazyk Perl. Naučit se pracovat s awkem je ale mnohem snazší.
Jak Perl, tak awk mají pověst nesrozumitelných jazyků, a to i pro samotné autory progra-mů – s oblibou se tvrdí, že jde o tzv. „write only“ jazyky. Nezapomínejte proto kód komen-tovat!

Cvičení

Několik praktických příkladů, kdy může být awk užitečný:

1. Pro účely prvního příkladu máme vstupní data formátována takto:

   uživatelské_jméno:*křestní_jméno*:*příjmení*:*telefonní_číslo*

   Vytvořte skript pro awk, který tento text zkonvertuje na záznamy pro databázi LDAP, která používá následující formát:

   dn: uid=*uživatelské_jméno*, dc=spolecnost, dc=czcn: *křestní_jméno příjmení*sn: *příjmení*telephoneNumber: *telefonní_číslo*

   Vytvořte si vstupní soubor s několika testovacími záznamy a zkuste provést konverzi.

2. Vytvořte skript bashe, který pomocí awk a standardních unixových příkazů nalezne tři uži-vatele s největším obsazením adresáře /home. (Pokud na počítači nemáte více uživatelů s vlastními domovskými adresáři, hledejte tři největší adresáře v kořenovém adresáři.) Nej-prve vše proveďte z příkazového řádku, pak vytvořte skript. Skript by měl poskytovat rozumně formátovaný výstup. Jakmile bude všechno fungovat, nechte si výsledky poslat e-mailem (například mail -s *Vyuziti disku* *<ja@moje_adresa>* < *výsledek*).

Jestliže používáte diskové kvóty, použijte jimi poskytované údaje. Jinak použijte příkaz find.

Ze vstupu odděleného tabelátory vytvořte výstup v XML stylu. Vstup vypadá takto:

slovo Strašně dlouhý řádek se spoustou popisu jinéslovo další dlouhý řádek slovo2 opět dlouhý řádek dalšíslovo děěsně dlouha-táááááááááááááááánský řádek, ale opravdu hooooooooooooooooooooooooodně moc

Výstup by měl vypadat takto:

<row>
<entry>slovo</entry>
<entry>
Strašně dlouhý řádek se spoustou popisu
</entry>
</row>
<row>
<entry>jinéslovo</entry>
<entry>
další dlouhý řádek
</entry>
</row>
<row>
<entry>slovo2</entry>
<entry>
opět dlouhý řádek
</entry>
</row>
<row>

<entry>dalšíslovo</entry>
<entry>
děěsně dlouhatáááááááááááá
áánský řádek, ale opravdu hooooooooooooooooooooooooodně moc
</entry>
</row>

Pokud něco víte o XML, doplňte skript o příkazy BEGIN a END a správně doformátujte soubor – anebo to udělejte v HTML.

Podmíněné příkazy

V této kapitole se budeme zabývat tím, jak se v bashi používají podmínky a podmíněné příkazy. Budeme hovořit o následujících tématech:

Příkaz if
Použití návratového kódu programu

Porovnání a testování vstupu a souborů
Konstrukce if/then/else
Konstrukce if/then/elif/else
Použití a testování pozičních parametrů
Vnořené příkazy if
Booleovské výrazy
Použití příkazu case

Úvod k příkazu if

Obecné

V některých situacích potřebujete zajistit, aby skript dělal různé činnosti podle toho, zda nějakýpříkaz uspěl či neuspěl. V těchto případech se používá konstrukce if.Nejjednodušší syntaxe příkazu if vypadá takto:

if TESTOVACÍ-PŘÍKAZY; then NÁSLEDNÉ-PŘÍKAZY; fi

Nejprve se provede sekvence testovacích příkazů, a pokud je její návratový kód nulový, provede se sekvence následných příkazů. Návratová hodnota celé konstrukce je kód posledního provedeného příkazu nebo nula, pokud nebyla podmínka vyhodnocena úspěšně.

Testovací příkazy často zahrnují různá porovnání řetězců nebo čísel, můžete však použít jakýko-liv příkaz, který při úspěšném proběhnutí vrací nulu a jinak nenulovou hodnotu. Pomocí unárních výrazů se často testují stavy souborů. Pokud jako název souboru v některém z primárních výrazů použijete zápis /dev/fd/N, bude se testovat souborový deskriptor „N“, testovat lze také stdin, stdout a stderr a jejich odpovídající deskriptory.

Výrazy používané v příkazu if

Následující tabulka uvádí seznam takzvaných primárních výrazů, z nichž se často vytváří testova-cí příkazy či seznamy testovacích příkazů. Primární výrazy se uvádějí v hranatých závorkách, které indikují, že jde o testování podmínky.
Primární výraz Význam

[-aSOUBOR] Splněno, pokud SOUBORexistuje. [-bSOUBOR] Splněno, pokud SOUBORexistuje a jde o speciální blokový soubor. [-cSOUBOR] Splněno, pokud SOUBORexistuje a jde o speciální znakový soubor. [-dSOUBOR] Splněno, pokud SOUBORexistuje a jde o adresář. [-eSOUBOR] Splněno, pokud SOUBORexistuje. [-fSOUBOR] Splněno, pokud SOUBORexistuje a jde o normální soubor. [-gSOUBOR] Splněno, pokud SOUBORexistuje a má nastaven SGID příznak. [-hSOUBOR] Splněno, pokud SOUBORexistuje a jde o symbolický odkaz. [-kSOUBOR] Splněno, pokud SOUBORexistuje a má nastaven sticky příznak. [-pSOUBOR] Splněno, pokud SOUBORexistuje a jde o pojmenovanou rouru (FIFO). [-rSOUBOR] Splněno, pokud SOUBORexistuje a lze jej číst. [-sSOUBOR] Splněno, pokud SOUBORexistuje a má nenulovou velikost. [-tFD] Splněno, pokud je souborový deskriptor FDotevřený a míří na terminál. [-uSOUBOR] Splněno, pokud SOUBORexistuje a má nastaven příznak SUID. [-wSOUBOR] Splněno, pokud SOUBORsoubor existuje a je zapisovatelný. [-xSOUBOR] Splněno, pokud SOUBORexistuje a je spustitelný. [-OSOUBOR] Splněno, pokud SOUBORexistuje a je vlastněn uživatelem s aktuálním efektivním ID. [-GSOUBOR] Splněno, pokud SOUBORexistuje a je vlastněn skupinou s aktuálním efektivním ID. [-LSOUBOR] Splněno, pokud SOUBORexistuje a jde o symbolický odkaz. [-NSOUBOR] Splněno, pokud SOUBORexistuje a byl od posledního čtení změněn. [-SSOUBOR] Splněno, pokud SOUBORexistuje a jde o soket. [SOUBOR1-ntSOUBOR2] Splněno, pokud byl SOUBOR1změněn později než SOUBOR2nebo pokud
SOUBOR1existuje a SOUBOR2ne. [SOUBOR1-otSOUBOR2] Splněno, pokud je SOUBOR1starší než SOUBOR2nebo pokud SOUBOR2existuje a SOUBOR1ne. [SOUBOR1-efSOUBOR2] Splněno, pokud se SOUBOR1i SOUBOR2odkazují na stejné zařízení a stejné číslo
inode. [-oNÁZEV_VOLBY] Splněno, pokud je zapnuta volba shellu "NÁZEV_VOLBY". [-zŘETĚZEC] Splněno, má-li ŘETĚZECnulovou délku. [-nŘETĚZEC]nebo [ŘETĚZEC]Splněno, má-li ŘETĚZECnenulovou délku. [ŘETĚZEC1== ŘETĚZEC2] Splněno, jsou-li řetězce shodné, pro striktní kompatibilitu s normou POSIX je možno
namísto "==" použít operátor "=". [ŘETĚZEC1!= ŘETĚZEC2] Splněno, pokud řetězce nejsou shodné. [ŘETĚZEC1< ŘETĚZEC2] Splněno, pokud je při lexikografickém řazení dle aktuálního lokalizačního nastavení
řazen ŘETĚZEC1před ŘETĚZEC2. [ŘETĚZEC1> ŘETĚZEC2] Splněno, pokud je při lexikografickém řazení dle aktuálního lokalizačního nastavení řazen ŘETĚZEC1za ŘETĚZEC2. [ARG1OP ARG2] "OP" je jedno z -eq, -ne, -lt, -le, -gtnebo -ge. Tyto aritmetické binární operá-tory vracejí true v případě, že je ARG1roven,

neroven, menší než, menší nebo roven než, větší než, respektive větší nebo roven než ARG2. Hodnoty ARG1a ARG2jsou číselné.

Primární výrazy
Jednotlivé výrazy je možno kombinovat pomocí následujících operátorů, uvedených v pořadí klesající priority:

Operátor	Popis
EXPR	Negace podmínky.
EXPR	Vrací hodnotu podmínky, používá se ke změně priority operátorů.
EXPR1 EXPR2	Platí, pokud jsou splněny obě podmínky.
EXPR1 EXPR2	Platí, pokud je splněna alespoň jedna podmínka.

– Kombinování výrazů.

Vestavěný příkaz [(nebo test) vyhodnocuje podmíněné výrazy pomocí pravidel vycházejících z počtu parametrů. Podrobnější informace o tomto tématu naleznete v dokumentaci k bashi. Stej-ně jako je nutné příkaz if uzavřít uzavíracím příkazem fi, je nutné i testovací příkaz (levá hrana-tá závorka) po vypsání celé podmínky uzavřít uzavírací pravou hranatou závorkou.

Příkazy ve větvi then

Seznam následných příkazů za klíčovým slovem then může být posloupnost jakýchkoliv platných uni-xových příkazů, spustitelných programů, shellových skriptů nebo příkazů shellu, s výjimkou uzavíra-cího příkazu fi. Je nutné mít na paměti, že shell chápe klíčová slova then a fi jako samostatné příka-zy. Proto je nutné je při zápisu na příkazovém řádku oddělovat středníkem, viz následující příklad.

Příklad

Ve skriptech bývají jednotlivé části konstrukce if obvykle opticky názorně odděleny, jak ukazuje několik následujících jednoduchých příkladů.

Testování souborů

První příklad testuje, zda soubor existuje:

```
[jura@jura tldp]$ cat msgcheck.sh#!/bin/bash

echo "Tento skript ověřuje, zda existuje systémový log."
echo "Ověřuji..."
if [ -f /var/log/messages ]; then

      echo "Soubor /var/log/messages existuje." fi echo echo "...hotovo."

[jura@jura tldp]$ ./msgcheck.shTento skript ověřuje, zda existuje systémový log.Ověřuji...Soubor /var/log/messages existuje.

...hotovo.
```

Testování voleb shellu

Můžete si přidat do konfiguračních souborů shellu:

```
# Tyto řádky testují, zda je nastavena volba noclobber:

if [ -o noclobber ] then
        echo “Soubory jsou chráněny před neúmyslným přepsáním při přesměrová
ní.” fi
```

Prostředí

Výše uvedený příklad můžete použít i na příkazovém řádku:

```
[jura@jura tldp]$ if [ -o noclobber ] ; then echo ; echo "Soubory jsou chráněny před neúmyslným přepsáním při přesměrování." ; echo ; fi
[jura@jura tldp]$ set -o noclobber [jura@jura tldp]$ if [ -o noclobber ] ; then echo ; echo "Soubory jsou chráněny před neúmyslným přepsáním při
přesměrování." ; echo ; fi
```

Soubory jsou chráněny před neúmyslným přepsáním při přesměrování.

> Pokud testujete nastavení prostředí, můžete na příkazovém řádku a ve skriptech dostávat rozdílné výsledky, protože skript se provádí v novém shellu, ve kterém nemusí být oče-kávané proměnné a volby automaticky nastaveny.

Jednoduchá použití příkazu if Testování návratového kódu

Proměnná ? obsahuje návratový kód předchozího provedeného příkazu (posledního ukončenéhoprocesu běžícího na popředí).Následující příklad ukazuje jednoduchý test:

```
[jura@jura tldp]$ if [ $? -eq 0 ]> then echo 'V pořádku!'> fiV pořádku!
```

Následující příklad ukazuje, že jako testovací příkaz lze v podmínce použít jakýkoliv unixový pří-kaz, který vrací návratový kód, a že příkaz if samotný vrací nulový návratový kód:

```
[jura@jura tldp]$ if grep $USER /etc/passwd; then echo "Váš uživatelský účet je spravován lokálně"; fi
jura:x:500:500::/home/jura:/bin/bash Váš uživatelský účet je spravován lokálně
[jura@jura tldp]$ echo $?
```

Stejný výsledek můžeme získat i takto:

```
[jura@jura tldp]$ grep $USER /etc/passwd jura:x:500:500::/home/jura:/bin/bash [jura@jura tldp]$ if [ $? -eq 0 ]; then echo "lokální účet"; fi
lokální účet
```

Porovnávání číselných hodnot

Následující příklad ukazuje porovnávání číselných hodnot:

```
[jura@jura tldp]$ num=`wc -l prace.txt` [jura@jura tldp]$ echo $num

[jura@jura tldp]$ if [ "$num" -gt "150" ] > then echo; echo "Dneska už jsi pracoval dost." > echo;
fi

Dneska už jsi pracoval dost.
```

Následující skript se spouští z cronu každou neděli. Je-li číslo týdne sudé, připomene vám, že máte vytáhnout popelnice před dům:

```
#!/bin/bash

# Vypocet cisla tydne prikazem date:

WEEKOFFSET=$[ $(date +"%V") % 2 ]

# Testujeme zbytek. Je-li nulovy, tyden je sudy a posilame upozorneni. # Jinak nedelame nic.

if [ $WEEKOFFSET -eq "0" ]; then echo "Je neděle večer, vytáhni popelnici před dům." | mail -s
    "Vytáhnout popelnici!" your@your_domain.org
```

Porovnávání řetězců

Pomocí porovnání řetězců ověřujeme identitu uživatele:

```
if [ "$(whoami)" != 'root' ]; then echo "Nejsi root, nemůžeš spustit $0." exit 1;
fi
```

V bashi můžete tuto konstrukci zapsat i kratším způsobem. Kompaktnější zápis testu by vypadal takto:

```
[ “$(whoami)” != ‘root’ ] && ( echo Používáš neprivilegovaný účet; exit 1 )
```

Operátor „&&“ definuje, co se má provést v případě splnění podmínky, analogicky lze operáto-rem „||“ definovat akci při nesplnění podmínky.Při porovnávání je možné použít i regulární výrazy:

```
[jura@jura tldp]$ if [[ "$pohlavi" == z* ]] > then echo "Těší mě, madame."; fi Těší mě,
madame.
```

> Opravdoví programátoři Většina programátorů dává přednost vestavěnému příkazu test, který je ekvivalentní hra-natým závorkám:

```
test "$(whoami)" != 'root' && (echo Používáš neprivilegovaný účet; exit 1)
```

Složitější použití příkazu if

Konstrukce if/then/else Hloupý příklad

Následující příklad ukazuje, jak provést jednu operaci, je-li podmínka if splněna, a jinou operaci, pokud podmínka splněna není.

[jura@jura tldp]$ pohlavi="žena"[jura@jura tldp]$ if [["$pohlavi" == "m*"]]> then echo "Dobrý den, pane"> else echo "Dobrý den, paní"> fiDobrý den, paní

V „seznamu alternativních následných příkazů“ ve větvi else mohou být, stejně jako v „seznamunásledných příkazů“ ve větvi then, použity libovolné unixové příkazy.Ještě jiný příklad, kterým rozšiřujeme příklad z kapitoly „Testování návratového kódu“:

[jura@jv2 ~]$ if ! grep ^$USER /etc/passwd 1> /dev/null > then echo "Váš účet není spravován lokálně" > else echo "Váš účet je spravován lokálně v souboru /etc/passwd" > fi Váš účet je spravován lokálně v souboru /etc/passwd

Ověřování parametrů předávaných na příkazovém řádku

Namísto nastavení hodnoty nějaké proměnné a následného spuštění skriptu je mnohem elegant-

nější a častěji používaný způsob předávání hodnot proměnných na příkazovém řádku.K tomuto účelu slouží poziční parametry $1, $2, ..., $N, proměnná $# obsahuje počet parametrůna příkazovém řádku a proměnná $0 samotný název spuštěného skriptu. Jednoduchý příklad vypadá takto:

A takto vypadá další příklad, který používá dva parametry:

```
[jura@jv2 tldp]$ cat vaha.sh #!/bin/bash

# Tento skript ze zadané hmotnosti v kg a výšky v cm vypočítá, # jak na tom jste.

vaha="$1"
vyska="$2"
idealnivaha=$[$vyska - 110]


if [ $vaha -le $idealnivaha ] ; then
    echo "Měl(a) byste jíst víc masa." else
    echo "Měl(a) byste jíst víc zeleniny." fi

[jura@jv2 tldp]$ bash -x vaha.sh 90 190
+ vaha=90
+ vyska=190
+ idealnivaha=80
+ '[' 90 -le 80 ']'
+ echo 'Měl(a) byste jíst víc zeleniny.' Měl(a) byste jíst víc zeleniny.
```

Testování počtu parametrů

Následující příklad je vylepšením předchozího příkladu o kontrolu, zda byl skript spuštěn právě se dvěma parametry:

```
[jura@jv2 tldp]$ cat vaha.sh #!/bin/bash

# Tento skript ze zadané hmotnosti v kg a výšky v cm vypočítá, # jak na tom jste.

if [ ! $# == 2 ]; then
    echo "Návod: $0 hmotnost_v_kg výška_v_cm"
    exit fi

vaha="$1"
vyska="$2"
idealnivaha=$[$vyska - 110]


if [ $vaha -le $idealnivaha ] ; then
    echo "Měl(a) byste jíst víc masa." else
```

```
    echo "Měl(a) byste jíst víc zeleniny." fi

[jura@jv2 tldp]$ ./vaha.sh 90 190 Měl(a) byste jíst víc zeleniny.
[jura@jv2 tldp]$ ./vaha.sh 90 190 33 Návod: ./vaha.sh hmotnost_v_kg výška_v_cm
```

Na hodnotu prvního parametru se odkazujeme proměnnou $1, na hodnotu druhého proměnnou $2 a tak dále. Celkový počet parametrů je uložen v proměnné $#.V kapitole „Použití příkazů if a exit“ si ukážeme elegantnější způsob, jak vypsat návod k použitískriptu.

Ověření existence souboru

Následující test se používá v celé řadě skriptů, protože v celé řadě případů nemá smysl skript provádět, pokud nejsou splněny potřebné podmínky pro jeho správné vykonání.

```
#!/bin/bash

# Tento skript poskytuje informace o souboru.

FILENAME="$1"

echo "Vlastnosti souboru $FILENAME:"

if [ -f $FILENAME ]; then
    echo "Velikost: $(ls -lh $FILENAME | awk '{ print $5 }')"
    echo "Typ: $(file $FILENAME | cut -d":" -f2 -)"
    echo "Číslo inode: $(ls -i $FILENAME | cut -d" " -f1 -)"
    echo "$(df -h $FILENAME | grep -v Mounted | awk '{ print "Na zařízení",$1", \ které je připojeno jako oddíl",$6,".'}')" else
    echo "Soubor neexistuje." fi
```

Všimněte si, že název souboru po celou dobu čteme z proměnné, kterou v našem případě napl-níme hodnotou prvního parametru skriptu. Jestliže se název souboru nepředává na příkazovém řádku, typicky se definuje jako „konstanta“ někde na začátku skriptu a v dalším těle se opět pra-cuje jen s touto proměnnou. Pokud byste někdy potřebovali název souboru změnit, stačí to pak provést jen na jediném místě.

Konstrukce if/then/elif/else Obecné

Tato konstrukce představuje úplnou podobu příkazu if:

```
if TESTOVACÍ_PŘÍKAZY; then
NÁSLEDNÉ_PŘÍKAZY;
elif DALŠÍ_TESTOVACÍ_PŘÍKAZY; then
DALŠÍ_NÁSLEDNÉ_PŘÍKAZY;
...
else ALTERNATIVNÍ_NÁSLEDNÉ_PŘÍKAZY;
fi
```

Nejprve se provede seznam testovacích_příkazů, a pokud je jeho návratový kód nulový, provede se seznam následných_příkazů. Pokud testovací_příkazy vrátí nenulový návratový kód, zpracová-vají se další_testovací_příkazy v jednotlivých větvích elif. Jestliže kterýkoliv seznam dalších_testo-vacích_příkazů vrátí nulový návratový kód, provede se odpovídající seznam dalších_násled-ných_příkazů a celý příkaz končí. Je-li definován seznam alternativních_následných_příkazů a všechny testy ve větvi if i ve všech větvích elif skončí neúspěšně, provede se tento alternativní seznam. Návratový kód celého příkazu bude roven návratovému kódu posledního provedeného příkazu nebo bude nulový, pokud nebyla splněna žádná podmínka.

Příklad

Následující příklad můžete každý den spouštět ze svého crontabu:

```
#!/bin/bash

# Tento skript provádí jednoduchou kontrolu volného místa

space=`df -h | awk '{print $5}' | grep % | grep -v Use | sort -n | tail -1 |
cut -d "%" -f1 -`
alertvalue="80"
```

```
if [ "$space" -ge "$alertvalue" ]; then

    echo "Nejméně jeden disk je skoro plný!" | mail -s "denní kontrola disků" root else
    echo "Obsazení disků je v pořádku" | mail -s "denní kontrola disků" root fi
```

Vnořené příkazy if

Uvnitř příkazu if můžete použít další příkaz if. Počet úrovní vnoření není omezen. Následující příklad testuje, zda je rok přestupný:

```
[jura@jv2 tldp]$ cat testleap.sh #!/bin/bash # Tento skript testuje, zda je rok přestupný.

year=`date +%Y`

        if [ $[$year % 400] -eq "0" ]; then echo "Tento rok je přestupný, únor
           má 29 dní." elif [ $[$year % 4] -eq 0 ]; then if [ $[$year % 100] -ne
           0 ]; then echo "Tento rok je přestupný, únor má 29 dní." else echo
           "Tento rok není přestupný, únor má 28 dní." fi else echo "Tento rok
           není přestupný, únor má 28 dní." fi

[jura@jv2 tldp]$ dateNe zář 17 10:30:00 CEST 2006

[jura@jv2 tldp]$ ./testleap.shTento rok není přestupný, únor má 28 dní.
```

Booleovské operace

Výše uvedený skript je možné zkrátit pomocí booleovských operátorů „AND“ (&&) a „OR“ (||).

Při testování aritmetických výrazů používáme zdvojené závorky, viz kapitolu „Aritmetická expan-ze“. Tento zápis je ekvivalentní příkazu let. Pokud byste zde chtěli používat hranaté závorky, například něco jako $[$year % 400], nedopadlo by to dobře, protože v této situaci hranaté závor-ky nereprezentují příkaz.

Editor gvim je jeden z těch editorů, které podle typu souboru volí barevné schéma zvýrazňování syntaxe. Díky tomu se v těchto editorech mnohem snáze hledají chyby.

Použití příkazů if a exit

S příkazem exit jsme se už letmo potkali v kapitole „Testování počtu parametrů“. Ukončuje pro-vádění celého skriptu. Nejčastěji se používá v případech, kdy uživatel nezadal potřebný vstup, nějaký příkaz neproběhl úspěšně nebo došlo k nějaké jiné chybě.

Příkaz exit pracuje s jedním nepovinným parametrem. Tím je celočíselný návratový kód, který bude předán rodiči skriptu a uložen v proměnné $?. Nulový návratový kód znamená, že skript proběhl úspěšně. Jakoukoliv jinou hodnotou může pro-gramátor rodičovskému procesu signalizovat, že došlo k nějaké chybě, a rodič na ni může ade-kvátně zareagovat. Pokud příkazu exit nezadáte žádný parametr, bude rodiči vrácena aktuální hodnota proměnné $?.

Následující příklad ukazuje mírně upravený skript tucnak.sh, který vrací návratový kód svému rodiči, krmeni.sh:

```
[jura@jv2 tldp]$ cat tucnak.sh #!/bin/bash

# Tento skript umožňuje krmit Tuxe. Tux má rád jenom ryby. Kromě toho jsme # přidali delfína a (asi) velblouda.

if [ "$menu" == "ryba" ]; then if [ "$zvire" == "tucnak" ]; then
        echo "Já rybu rád, já rybu moc rád!" elif [ "$zvire" ==
     "delfin" ]; then echo "Víííípíííífíííí!" else echo
     "*prrrrrrrt*" fi else
    if [ "$zvire" == "tucnak" ]; then
        echo "Tohle Tux nemá rád! Tux chce rybu!"
        exit 1

    elif [ "$zvire" == "delfín" ]; then
        echo "VuííííííVíííííí!"
        exit 2

    else
        echo "Umíš číst cedulky?!"
        exit 3
```

```
fi fi
```

Tento skript budeme volat z následujícího skriptu, ve kterém proto exportujeme proměnné menu a zvire:

```
[jura@jv2 tldp]$ cat krmeni.sh #!/bin/bash # Skript reaguje na návratový kód skriptu tucnak.sh

export menu="$1" export zvire="$2"

stravnik="./tucnak.sh"

$stravnik $menu $zvire

case $? in

1) echo "Hlídač: Radši jim daj rybu, jinak budou hrozně vzteklí..." ;;
2) echo "Hlídač: Přesně kvůli lidem jako vy opouštějí naši planetu..." ;;
3) echo "Hlídač: Můžete ho sponzorovat, z čeho si myslíte, že bude živý?" ;;
*) echo "Hlídač: Nezapomeňte si tady průvodce!" ;;
esac


[jura@jv2 tldp]$ ./krmeni.sh jabko tucnakTohle Tux nemá rád! Tux chce rybu!Hlídač: Radši jim daj rybu, jinak budou hrozně vzteklí...
```

Jak vidíte, návratové kódy je možno volit libovolně. Různé příkazy systému používají své definované hodnoty návratových kódů, které se dozvíte například na manuálových stránkách.

Použití příkazu case

Zjednodušení podmínek

Vnořené příkazy if jsou sice pěkná věc, jakmile se ale máte rozhodovat mezi více možnými akce-mi, mohou vznikat nepřehledné konstrukce. Pro složitější případy je vhodnější příkaz case:

```
case V›RAZ in MOŽNOST1) PŘÍKAZY;; MOŽNOST2) PŘÍKAZY;; ... MOŽNOSTN) PŘÍKAZY;; esac
```

Každá možnost představuje vzor, kterému může odpovídat vyhodnocený výraz. Provedou se pří-kazy první vyhovující možnosti. Operátorem „|“ je možno oddělit více možností, operátor „)“ ukončuje seznam možností. Jednotlivé možnosti a jim odpovídající příkazy označujeme jako *větve*. Každá větev musí končit znaky „;;“. Celý příkaz case je ukončen příkazem esac.

V následujícím příkladu pomocí příkazu case generujeme podrobněji rozdělená hlášení o využití diskového prostoru:

```
[jura@jv2 tldp]$ cat disktest.sh #!/bin/bash

# Tento skript provádí jednoduchou kontrolu volného místa

space=`df -h | awk '{print $5}' | grep % | grep -v Use | sort -n | tail -1 | cut -d "%" -f1 -`

case $space in
[1-6]*) Message="Všechno je v pořádku." ;;
  [7-8]*) Message="Začni přemýšlet nad úklidem disku. Máme
     svazek zaplněný na $space %." ;;
9[1-8]) Message="Začni shánět nový disk! Máme svazek zaplněný na $space %!" ;;
99) Message="To je konec! Svazek je zaplněn na $space %!" ;;
*) Message="Vypadá to na nesmyslný údaj o zaplnění disku..." ;;
esac


echo $Message | mail -s "zprava o zaplneni disku `date`" jura


[jura@jv2 tldp]$
You have mail in /var/spool/mail/jura


[jura@jv2 tldp]$ tail -16 /var/spool/mail/juraFrom jura@localhost.localdomain Sun Sep 17 11:19:35 2006Return-Path:
<jura@localhost.localdomain>Received: from localhost.localdomain (localhost.localdomain [127.0.0.1])
        by localhost.localdomain (8.13.1/8.13.1) with ESMTP id k8H9JZNr007738 for <jura@localhost.localdomain>; Sun, 17 Sep 2006
```

```
        11:19:35 +0200

Received: (from jura@localhost) by localhost.localdomain (8.13.1/8.13.1/Submit) id k8H9JZxA007735 for jura; Sun, 17 Sep 2006 11:19:35
        +0200
Date: Sun, 17 Sep 2006 11:19:35 +0200 From: jura@localhost.localdomain Message-Id:
<200609170919.k8H9JZxA007735@localhost.localdomain> To: jura@localhost.localdomain Subject: zprava o zaplneni disku Ne zá 17 11:19:35
CEST 2006

Všechno je v pořádku.
```

Výsledek běhu skriptu si samozřejmě můžete přečíst přímo ve svém poštovním klientovi. Příklad

pouze ukazuje, že skript odešle korektní e-mail s údaji „To:“, „Subject:“ a „From:“.Celou řadu dalších příkladů použití příkazu case naleznete v adresáři s inicializačními skriptysystému. Tyto skripty typicky reagují na hodnoty start a stop a spouštějí či zastavují systémovéprocesy. Teoretický příklad naleznete v následující kapitole.

Příklad inicializačního skriptu

Inicializační skripty velmi často používají příkaz case a na základě jeho vyhodnocení spouštějí, zastavují či zjišťují stav systémových služeb. Následující příklad pochází ze skriptu služby Anacron, démona, který opakovaně spouští příkazy s periodou v řádu dnů.

```
case "$1" in
        start)
                start
                ;;


        stop)
                stop
                ;;


        status)
                status anacron
                ;;

        restart)
                stop
                start
                ;;

        condrestart)
                if test "x`pidof anacron`" != x; then
                        stop
                        start

                fi
                ;;


        *) echo $"Usage: $0 {start|stop|restart|condrestart|status}" exit 1

esac
```

Úkony prováděné v jednotlivých větvích, jako je například zastavení nebo spuštění démona, jsou definovány ve funkcích, které jsou z části definovány v souboru /etc/rc.d/init.d/functions. Další podrobnosti se dozvíme v kapitole „Funkce“.

Shrnutí

V této kapitole jsme se dozvěděli, jak ve skriptech pracovat s podmínkami, takže podle úspěchu či neúspěchu různých příkazů můžeme provádět různé operace. Testování se provádí příkazem if. Je možné se rozhodovat na základě aritmetického či řetězcového porovnání, na základě návra-tového kódu nebo podle přítomnosti či nepřítomnosti vstupních parametrů či souborů.

Příkazy if/then/fi se často používají k potlačení výstupu skriptu, takže je možné skript snadno spouštět na pozadí nebo pomocí démona cron. Složitější podmínky se často řeší příkazem case. V případě úspěšného provedení

skriptu by měl být jeho rodič explicitně informován nulovým návratovým kódem. Při chybě je možné vracet libovolný nenulový kód. Podle jeho hodnoty může rodičovský program provést další potřebné akce.

Cvičení

Dále předkládáme několik nápadů, jak můžete s příkazem if ve skriptech pracovat:

Pomocí konstrukce if/then/elif/else vypište informace o kalendářním měsíci. Skript by měl vypsat počet dní v měsíci, v případě února by měl správně detekovat přestupný rok.
Vyzkoušejte to samé s příkazem case a alternativním použitím příkazu date.
Upravte soubor /etc/profile tak, abyste při přihlášení jako *root* dostali varovné hlášení.
Upravte skript leaptest.sh z kapitoly „Booleovské operace“, aby pracoval s jedním para-metrem, testovaným rokem. Kontrolujte, zda byl spuštěn právě s jedním parametrem.
Vytvořte skript whichdaemon.sh, který ověří, že na vašem systému běží démony httpd a init. Pokud běží httpd, vypište hlášení typu „na počítači běží webový server". Běžící pro-cesy zjistěte příkazem ps.
Vytvořte skript, který příkazem scp vytvoří zálohu vašeho domovského adresáře na vzdá-lený počítač. Skript by měl zapsat zprávu o svém průběhu například do souboru ~/log/homebackup.log. Nemáte-li k dispozici druhý počítač, kopírujte data příkazem scp na místní počítač. Ke kopírování potřebujete mít vytvořeny klíče, jinak byste museli zadá-vat heslo. Vytváření SSH klíčů je popsáno na manuálové stránce man *ssh-keygen*.

> K vytvoření zálohy použijte příkaz tar cf, pomocí příkazů gzip nebo bzip2 archiv zkom-primujte. Názvy všech souborů definujte v proměnných. V proměnných definujte rovněž název vzdáleného serveru a vzdáleného adresáře. Díky tomu se vám ulehčí budoucí změny skriptu.
>
> Skript musí ověřit, zda komprimovaný archiv již neexistuje, a pokud ano, nejprve jej musí smazat. Jinak by se zbytečně vypisovala chybová hlášení. Dále by měl skript otestovat, zda je k dispozici dostatek volného místa. V jednom oka-mžiku budete mít na disku data v domovském adresáři, data v archivu a data v kompri-movaném archivu. Pokud nebude k dispozici dostatek místa, zapište chybu do logu a ukončete provádění skriptu. Vytvořené archivy musí skript před skončením po sobě smazat.

Tvorba interaktivních skriptů

V této kapitole si ukážeme, jak může skript interagovat s uživatelem:

Výpis srozumitelných zpráv a vysvětlení
Zachycení uživatelského vstupu
Požádání uživatele o zadání vstupu
Použití souborových deskriptorů ke čtení a zápisu z a do více souborů

Zobrazování hlášení

Interaktivně nebo ne?

Některé skripty fungují bez jakékoliv interakce s uživatelem. Mezi výhody neinteraktivních skriptů patří:

Skript proběhne pokaždé stejným definovaným způsobem.
Skript je možno spouštět na pozadí.

Řada skriptů nicméně od uživatele potřebuje nějaký vstup nebo uživatele při svém běhu o něčem informuje. Mezi výhody interaktivních skriptů patří:

Skript může být flexibilnější.
Uživatel může za běhu modifikovat chování skriptu nebo jej přimět dělat různé věci.
Skript může uživatele informovat o svém průběhu.

Při tvorbě interaktivních skriptů se neomezujte na komentáře v těle skriptu. Skript, který vypisuje na obrazovku vhodná hlášení, je

uživatelsky mnohem přívětivější a snáze se také odlaďuje. Skript sice může pracovat perfektně, pokud ale uživatele nebude o své činnosti přesně informovat, uživatel nebude spokojen. Nezapomínejte proto například na upozornění typu, že nyní musí uživatel počkat, než skript dokončí nějaký výpočet. Je-li to možné, sdělte uživateli, jak dlouho bude proces trvat. Pokud nějaká operace obecně bude trvat dlouho, zvažte implementaci nějakého indikátoru průběhu.

Jestliže žádáte uživatele o vstupní informace, vždy je lepší podrobněji specifikovat, co vlastně očekává
te. S tím rovněž souvisí kontrola zadaných vstupních údajů a případný výpis návodu k použití skriptu. Zprávy pro uživatele můžete v bashi vypisovat příkazy echo a printf. V této chvíli byste sice měli dobře znát přinejmenším příkaz echo, v následujících částech si nicméně ukážeme několik nových příkladů.

Použití vestavěného příkazu echo

Vestavěný příkaz echo vypíše zadané parametry, odděluje je mezerou a výpis ukončí znakem nového řádku. Jeho návratový kód je vždy nulový. Příkaz echo pracuje s dvojicí přepínačů: ▪-e: bude interpretovat obráceným lomítkem escapované znaky ▪-n: nevypíše se finální znak nového řádku

Jako příklad podrobnějších komentářů vylepšíme skripty krmeni.sh a tucnak.sh z kapitoly „Ověřování parametrů předávaných na příkazovém řádku“:

```
[jura@jv2 tldp]$ cat tucnak.sh #!/bin/bash # Tento skript umožňuje krmit Tuxe. Tux má rád jenom ryby. Kromě toho jsme # přidali delfína a (asi) velblouda.

 if [ "$menu" == "ryba" ]; then if [ "$zvire" == "tucnak" ]; then
     echo -e "Já rybu rád, já rybu moc rád!\n" elif [ "$zvire" ==
     "delfin" ]; then echo -e "\a\a\aVíííípíííííííí!\a\a\a\n" else echo
     -e "*prrrrrrrt*\n" fi else
if [ "$zvire" == "tucnak" ]; then
    echo -e "Tohle Tux nemá rád! Tux chce rybu!\n"
    exit 1

elif [ "$zvire" == "delfin" ]; then
    echo -e "\a\a\a\a\a\aVuíííííí Víííííí!\a\a\a\n"
    exit 2

else
    echo -e "Umíš číst cedulky?! "$zvire" se nesmí krmit!\n"
    exit 3

fi fi

[jura@jv2 tldp]$ cat krmeni.sh #!/bin/bash # Skript reaguje na návratový kód skriptu tucnak.sh

if [ "$#" != "2" ]; then echo -e "Krmíme takto:\t$0 jídlo zvíře\n" exit 1
else

    export menu="$1"
    export zvire="$2"


    echo -e "Zvířeti '$zvire' předkládáme '$menu'...\n"
    stravnik="./tucnak.sh"
    $stravnik $menu $zvire


    vysledek="$?"
echo -e "Krmení dokončeno.\n"

case "$vysledek" in

1) echo -e "Hlídač: \"Radši jim daj rybu, jinak budou hrozně vzteklí...\"\n" ;;
        2) echo -e "Hlídač: \"Přesně kvůli lidem jako vy opouštějí naši plane-
            tu...\"\n" ;;
3) echo -e "Hlídač: \"Správné krmení si můžete koupit u vchodu!\"\n" echo -e "Hlídač: \"Nechcete je přece otrávit, že?\"\n" ;;
*) echo -e "Hlídač: \"Nezapomeňte si tady průvodce!\"\n" ;;
esac
```

```
fi


echo "Konec..."
echo -e "\a\a\aDíky za návštěvu v naší ZOO, těšíme se na vás příště\n"


[jura@jv2 tldp]$ ./krmeni.sh jabko velbloudZvířeti 'velbloud' předkládáme 'jabko'...

Umíš číst cedulky?! velbloud se nesmí krmit!


Krmení dokončeno.


Hlídač: "Správné krmení si můžete koupit u vchodu!"


Hlídač: "Nechcete je přece otrávit, že?"


Konec...
Díky za návštěvu v naší ZOO, těšíme se na vás příště'


[jura@jv2 tldp]$ ./krmeni.sh jabkoKrmíme takto: ./krmeni.sh jídlo zvíře
```

O escapovaní znaků hovoříme podrobněji v kapitole „Escapování znaků“. Následující tabulka obsahuje seznam speciálních sekvencí, jimž příkaz echo rozumí: Podrobnější informace o příkazu printf a možnostech, jak se jím dá formátovat výstup, nalezne-te na informačních stránkách bashe.

Sekvence	Význam
\	Upozornění (zvonek).
\	Backspace.
\	Potlačí koncový nový řádek.
\	Escape.
\	Nová stránka (FF).

Sekvence	Význam
\	Nový řádek (LF).
\	Návrat vozíku (CR).
\	Horizontální tabulátor.
\	Vertikální tabulátor.
\\	Zpětné lomítko.
\	Osmibitový znak definovaný hodnotou v osmičkové soustavě.
\	Osmibitový znak definovaný hodnotou v desítkové soustavě.
\	Osmibitový znak definovaný hodnotou v šestnáctkové soustavě.

Řídící sekvence používané příkazem echo

Zachycení uživatelského vstupu

Vestavěný příkaz read

Vestavěný příkaz read je protějškem příkazů echo a printf. Jeho syntaxe vypadá takto:

read [volby] NÁZEV1 NÁZEV2 ... NÁZEVN

Dojde k načtení jednoho řádku buď ze standardního vstupu nebo ze souborového deskriptorudefinovaného hodnotou přepínače -u. První slovo na řádku bude přiřazeno do proměnné NÁZEV1,druhé slovo do proměnné NÁZEV2 a tak dále. Všechna zbývající slova a jejich oddělovače skončív proměnné NÁZEVN. Je-li slov méně než zadaných proměnných, budou nadbytečným proměn-ným přiřazeny prázdné hodnoty.

K rozdělení vstupu na jednotlivá slova se používá hodnota proměnné IFS, viz kapitolu „Dělenís lov“. Pomocí obráceného lomítka je možné potlačit speciální význam následujícího znaku, pří-padně dosáhnout pokračování vstupu na následujícím řádku.

Nezadáte-li žádnou proměnnou, bude vstup načten do proměnné REPLY.Návratový kód příkazu read je nula s výjimkou případů, kdy dojde k načtení znaku konce sou-
boru, vyprší časový limit příkazu read nebo je pomocí volby -u zadán neplatný souborovýdeskriptor.Příkaz read v bashi pracuje s následujícími volbami:

Volba	Význam
POLE	Jednotlivá slova budou přiřazena do prvků pole POLE, počínaje indexem 0. Před přiřazením budou stávající prvky pole zrušeny. Případné názvy proměnných na příkazovém řádku se igno-rují.
DELIM	Jako konec vstupního řádku bude brán první znak proměnné DELIM, nikoliv výchozí konec řádku. K načtení řádku se použije funkce .
NCHARS	Příkaz načte jen NCHARSvstupních znaků, nečeká na celý řádek.
PROMPT	Vypíše PROMPTbez ukončení řádku. Prompt se vypisuje jen při čtení vstupu z terminálu.
	Je-li zadána tato volba, obrácené lomítko nefunguje jako escapovací znak, chápe se jako nor mální vstupní znak. Konkrétně nebude možné pokračovat vstupem na dalším řádku.
	Tichý režim. Načítá-li se vstup z terminálu, zadávané znaky se nevypisují.
TIMEOUT	Pokud nebude vstup zadán do TIMEOUTsekund, příkaz skončí chybou. Tato volba je funkční jen při čtení vstupu z terminálu nebo roury.
FD	Čte vstup ze souborového deskriptoru FD.

Volby příkazu read

Následující jednoduchý příklad rozšiřuje skript leaptest.sh z předchozí kapitoly:

```
[jura@jv2 tldp]$ cat leaptest.sh #!/bin/bash # Tento skript testuje, zda je rok přestupný.

echo "Zadej rok (4 číslice) a [ENTER]:"

read year

if (( ("$year" % 400) == "0" )) || (( ("$year" % 4 == "0") && ("$year" % 100 !
= "0") )); then echo "Rok $year je přestupný rok." else echo "Rok $year není
přestupný rok." fi

[jura@jv2 tldp]$ ./leaptest.sh Zadej rok (4 číslice) a [ENTER]:
2000
Rok 2000 je přestupný rok.
```

Požádání uživatele o vstup

Následující příklad ukazuje, jak můžete pomocí promptu požádat uživatele o zadání vstupu:

```
[jura@jv2 tldp]$ cat friends.sh #!/bin/bash

# Tento skript aktualizuje knihu kontaktů.

friends="/var/tmp/jura/friends"

echo "Ahoj, "$USER". Tento skript tě zaregistruje v Jurově seznamu kamarádů."

echo -n "Zadej své jméno a stiskni [ENTER]: " read name echo -n "Zadej své pohlaví (m/ž) a stiskni [ENTER]: " read -n 1 gender echo
```

```
grep -i "$name" "$friends"
if [ $? == 0 ]; then echo "Už jsi zaregistrován(a), končím." exit 1
elif [ "$gender" == "m" ]; then echo "Byl jsi přidán do seznamu kontaktů." exit 1
else echo -n "Kolik je ti let? " read age if [ "$age" -lt "25" ]; then
        echo -n "Jakou máš barvu vlasů? "
        read colour
        echo "$name $age $colour" >> "$friends"
        echo "Byla jsi přidána do seznamu Jurových kamarádů! Díky moc!"

    else
        echo "Byla jsi přidána do seznamu Jurových kamarádů."
        exit 1

    fi fi
```

```
[jura@jv2 tldp]$ cp friends.sh /var/tmp

[jura@jv2 tldp]$ cd /var/tmp/; mkdir jura; cd jura; touch friends; chmod a+rw friends

[jura@jv2 jura]$ /var/tmp/friends.sh Ahoj, jura. Tento skript tě zaregistruje v Jurově seznamu kamarádů. Zadej své jméno a stiskni [ENTER]: jura Zadej své pohlaví (m/ž) a stiskni [ENTER]: m Byl jsi přidán do seznamu kontaktů.

[jura@jv2 jura]$ cat /var/tmp/jura/friends
```

Soubor je skutečně prázdný – skript ukládá informace jen o zajímavých objektech, i když pokaž-dé tvrdí, že kontakt zapsal.Teď mohou skript používat i jiní uživatelé:

```
[martina@jv2 ~]$ /var/tmp/friends.sh Ahoj, martina. Tento skript tě zaregistruje v Jurově seznamu kamarádů. Zadej své jméno a stiskni [ENTER]: Martina Zadej své pohlaví (m/ž) a stiskni [ENTER]: ž Kolik je ti let? 24 Jakou máš barvu vlasů? blond Byla jsi přidána do seznamu Jurových kamarádů! Díky moc!
```

Po nějaké době se soubor začne uspokojivě plnit:

```
Martina 24 blond Jana 18 hnědé Petra 21 blond
-kráceno -
```

Tato situace samozřejmě není ideální, protože soubor s kontakty může editovat kdokoliv (i když jej nemůže smazat). Problém byste mohli vyřešit speciálními přístupovými právy skriptu, viz kapi-tolu SUID a SGID v příručce „Úvod do Linuxu“.

Přesměrování a deskriptory souborů Obecné

Jak už víme ze základní práce se shellem, vstup a výstup příkazů je možno před jejich spuštěním přesměrovat pomocí speciálních znaků – operátorů přesměrování, které vyhodnocuje přímo shell. Pomocí přesměrování je možné v aktuálním prostředí shellu také otevírat a zavírat soubory.

Přesměrování je možné použít i ve skriptech, takže skript například může ze souboru načítat svůj vstup nebo do něj zapisovat výstup. Uživatel pak může s výstupním souborem dále pracovat, například jej předat ke zpracování jinému skriptu.

Vstup a výstup souborů se realizuje prostřednictvím celočíselných „handlů", které registrují všech-ny soubory otevřené daným procesem. Těmto číselným hodnotám se říká deskriptory souborů. Nejznámější deskriptory jsou *stdin*, *stdout* a *stderr*, kterým odpovídají hodnoty 0, 1 a 2. Tyto tři hodnoty a jim odpovídající zařízení jsou rezervovány. Bash dokáže jako souborový deskriptor použít i TCP nebo UDP porty na vzdáleném systému.

Následující výpis ukazuje, jak rezervované deskriptory ukazují na skutečná zařízení:

```
michel ~> ls -l /dev/std* lrwxrwxrwx 1 root root 17 Oct 2 07:46 /dev/stderr -> ../proc/self/fd/2 lrwxrwxrwx 1 root root 17 Oct 2 07:46 /dev/stdin -> ../proc/self/fd/0 lrwxrwxrwx 1 root root 17 Oct 2 07:46 /dev/stdout -> ../proc/self/fd/1

michel ~> ls -l /proc/self/fd/[0-2] lrwx------1 michel michel 64 Jan 23 12:11 /proc/self/fd/0 -> /dev/pts/6 lrwx------1 michel michel 64 Jan 23 12:11 /proc/self/fd/1 -> /dev/pts/6 lrwx------1 michel michel 64 Jan 23 12:11 /proc/self/fd/2 -> /dev/pts/6
```

Podrobnější informace o adresářích v souborovém systému /proc a o tom, jak systém pro jednt-livé procesy obsluhuje standardní souborové deskriptory, můžete najít na stránkách info MAKEDEV a info proc.

Když spustíte skript z příkazového řádku, nic zvláštního se nestane, protože synovský shell zdědí stejné souborové deskriptory, jako používá jeho rodič. Pokud rodič není k dispozici (například spouštíte-li skript prostřednictvím démona *cron*), standardní souborové deskriptory budou buď roury nebo nějaké dočasné soubory, pokud ovšem nenastavíte přesměrování jinak. Demonstruje to následující příklad, který ukazuje výstup jednoduchého skriptu pro příkaz at:

[jura@jv2 ~]$ date Ne zář 17 13:10:46 CEST 2006

[jura@jv2 ~]$ at 1313 at> ls -l /proc/self/fd/ > /var/tmp/fdtest.at at> <EOT> job 3 at 2006-09-17 13:13

[jura@jv2 ~]$ cat /var/tmp/fdtest.at celkem 4 lr-x------1 jura jura 64 zář 17 13:13 0 -> /tmp/sh-thd-1158498796 (deleted) l-wx------1 jura jura 64 zář 17 13:13 1 -> /var/tmp/fdtest.at l-wx------1 jura jura 64 zář 17 13:13 2 -> /var/spool/at/spool/a0000301269ea1 lr-x------1 jura jura 64 zář 17 13:13 3 -> /proc/7985/fd

A jeden příklad s démonem cron:

[jura@jv2 ~]$ crontab -l 17 13 * * * ls -l /proc/self/fd > /var/tmp/fdtest.cron

[jura@jv2 ~]$ cat /var/tmp/fdtest.cron total 4 lr-x------1 jura jura 64 Sep 17 13:17 0 -> pipe:[48743] l-wx------1 jura jura 64 Sep 17 13:17 1 -> /var/tmp/fdtest.cron l-wx------1 jura jura 64 Sep 17 13:17 2 -> pipe:[48744] lr-x------1 jura jura 64 Sep 17 13:17 3 -> /proc/7991/fd

Přesměrování chyb

Z předchozích příkladů je zřejmé, že skriptu můžete předat vstupní a výstupní soubory (podrob-nosti v kapitole „Vstup a výstup souborů“), občas se ale zapomíná na přesměrování chyb – což je výstup, který může být rovněž potřebný. Pokud budete mít štěstí, dojdou vám chyby e-mailem a z něj třeba příčinu chyby odhalíte. Pokud nebudete mít štěstí, skript v důsledku chyby selže a nikam se nic nepošle, takže bude následovat pracné ladění.

Při přesměrovávání chyb nezapomínejte, že je důležité pořadí přesměrování. Například následují-cí příkaz zadaný ve /var/spool:

ls -l * *2*> /var/tmp/unaccessible-in-spool

Chybový výstup příkazu ls bude přesměrován do souboru unaccessible-in-spool v adresáři /var/tmp. Tento příkaz:

ls -l * > /var/tmp/spoollist *2>&1*

přesměruje standardní výstup i standardní chybový výstup do souboru spoollist. A příkaz:

ls -l * *2>&1* > /var/tmp/spoollist

přesměruje do souboru jenom standardní výstup, protože standardní chybový výstup byl přesmě

rován na standardní výstup ještě před přesměrováním standardního výstupu.Pokud se ví, že chybové zprávy nebudou zapotřebí, velmi často se chybový výstup přesměrová-vá do /dev/null. Stovky příkladů naleznete v inicializačních skriptech systému.

Bash umožňuje přesměrovat standardní výstup i standardní chybový výstup následujícím zápisem:

&> SOUBORJde o ekvivalent zápisu > SOUBOR 2>&1, který jsme používali v předchozích příkladech. Velmičasto se používá při přesměrování do /dev/null v případech, že chcete příkaz jen spustit a vůbecvás nezajímá jeho výstup ani případné chyby.

Vstup a výstup souborů Použití /dev/fd

Adresář /dev/fd obsahuje položky pojmenované 0, 1, 2 a tak dále. Otevření souboru /dev/fd/N je ekvivalentní manipulaci se souborovým deskriptorem číslo *N*. Pokud ve svém systému máte zařízení /dev/stdin, /dev/stdout a /dev/stderr, jde o ekvivalenty položek /dev/fd/0, /dev/fd/1 a /dev/fd/2.

Nejčastěji se soubory z /dev/fd používají v shellu. Jde o mechanismus, díky němuž je možné se standardním vstupem a výstupem pracovat stejnými způsoby jako s běžnými soubory. Pokud váš systém nepodporuje zařízení /dev/fd, musíte problém nějak obejít. Často se k tomu používá pomlčka (-), která programu říká, že má vstup načítat z roury. Například:

michel ~> filter body.txt.gz | cat header.txt - footer.txt Tento text se vypisuje na začátku každé tiskové úlohy, děkujeme našim správcům za tak úžasné nastavení tiskové infrastruktury.

Text který prošel filtrem.

Tento text se vypisuje na konci každé tiskové úlohy.

Příkaz cat nejprve přečte soubor header.txt, pak čte svůj standardní vstup (což je výstup příka-zu filter) a nakonec soubor footer.txt. Pomlčka jako název souboru má speciální význam, sig-nalizuje použití standardního vstupu či výstupu, což je vlastnost, která bývá častou příčinou růz-ných chyb. Problémem může být také uvedení pomlčky jako prvního parametru, protože může být interpretována jako signalizace volby. Pomocí zařízení /dev/fd se dosáhne uniformity a pře-dejde se nejasnostem:

michel ~> filter body.txt | cat header.txt /dev/fd/0 footer.txt | lp

Veškerý výstup je v tomto příkladu navíc rourou předán příkazu lp, který jej vytiskne.

Read a exec

Přiřazení souborových deskriptorů souborům

Na souborové deskriptory je možno nahlížet i opačně, jako na mechanismus, který souborům při-řazuje čísla. Namísto toho, abyste pracovali s názvy souborů, můžete se odkazovat na příslušné deskriptory. K přiřazení deskriptoru danému souboru slouží vestavěný příkaz exec. Používá se takto:

exec fdN> soubor

Tímto zápisem bude výstup pro souborový deskriptor N přiřazen zadanému souboru.

exec fdN< soubor

A tímto zápisem bude vstup pro souborový deskriptor N přiřazen zadanému souboru.
Jakmile deskriptoru přiřadíte soubor, můžete jej používat pro přesměrování tak, jak to ukazuje následující příklad:

```
michel ~> exec 4 > result.txt

michel ~> filter body.txt | cat header.txt /dev/fd/0 footer.txt >& 4

michel ~> cat result.txt Tento text se vypisuje na začátku každé tiskové úlohy, děkujeme našim správcům za tak úžasné nastavení tiskové infrastruktury.

Text který prošel filtrem.

Tento text se vypisuje na konci každé tiskové úlohy.
```

Souborový deskriptor 5

Použití tohoto deskriptoru může vést k problémům, proto se jeho použití vyhýbejte.
Podrobnější informace k tomuto tématu naleznete v příručce Advanced Bash-Scripting Guide.

Čtení ve skriptech

Následující příklad ukazuje, jak je možné přecházet mezi vstupem ze souboru a vstupem z příkazového řádku:

```
[jura@jv2 tldp]$ cat sysnotes.sh#!/bin/bash

# Tento skript vytvoří seznam důležitých konfiguračních souborů, spojí
# je do jednoho záložního a umožní soubory okomentovat.


CONFIG=/var/tmp/sysconfig.out
rm "$CONFIG" 2>/dev/null


echo "Výstup bude uložen v $CONFIG."


exec 7<&0


exec < /etc/passwd


# Čteme první řádek /etc/passwd
read rootpasswd


echo "Ukládají se informace o účtu root..."echo "Informace o účtu root:" >>
"$CONFIG"echo $rootpasswd >> "$CONFIG"

exec 0<&7 7<&
```

```
echo -n "Zadejte komentář nebo [ENTER] pro prázdný komentář: "
read comment; echo $comment >> "$CONFIG"


echo "Ukládají se informace ze souboru hosts..."


# nejprve ze souboru hosts odstraníme komentáře
TEMP="/var/tmp/hosts.tmp"
cat /etc/hosts | grep -v "^#" > "$TEMP"


exec 7<&0
exec < "$TEMP"


read ip1 name1 alias1
read ip2 name2 alias2


echo "Soubor hosts:" >> "$CONFIG"


echo "$ip1 $name1 $alias1" >> "$CONFIG"
echo "$ip2 $name2 $alias2" >> "$CONFIG"

exec 0<&7 7<&


echo -n "Zadejte komentář nebo [ENTER] pro prázdný komentář: "
read comment; echo $comment >> "$CONFIG"
rm "$TEMP"


[jura@jv2 tldp]$ ./sysnotes.shVýstup bude uložen v /var/tmp/sysconfig.out.Ukládají se informace o účtu root...Zadejte komentář nebo [ENTER] pro prázdný komentář: heslo je takové to dlouhés čísílkama
Ukládají se informace ze souboru hosts...Zadejte komentář nebo [ENTER] pro prázdný komentář: všechno ostatní je v DNS

[jura@jv2 tldp]$ cat /var/tmp/sysconfig.outInformace o účtu root:root:x:
0:0:root:/root:/bin/bashheslo je takové to dlouhé s čísílkamaSoubor hosts:
127.0.0.1 localhost.localdomain localhost jv2
62.129.60.34 jv všechno ostatní je v DNS
```

Zavření souborových deskriptorů

Protože synovské procesy dědí otevřené souborové deskriptory, je vhodné deskriptory zavírat ve chvíli, kdy už nebudou dále zapotřebí. Provede se to příkazem:

exec fd<&

V předcházejícím příkladu zavíráme souborový deskriptor 7, přiřazený standardnímu vstupu, pokaždé když uživatel potřebuje pracovat se standardním vstupním zařízením, typicky klávesnicí. Následující příklad ukazuje jednoduché přesměrování standardního chybového výstupu do roury:

```
michel ~> cat listdirs.sh #!/bin/bash

# Tento skript nemění standardní výstup, přesměrovává však standardní # chybový výstup na příkaz awk.

INPUTDIR="$1"

exec 6>&1

ls "$INPUTDIR"/* 2>&1 >&6 6>&- \ # Zavíráme fd 6 pro awk, nikoliv však pro ls.

| awk 'BEGIN { FS=":" } { print "YOU HAVE NO ACCESS TO" $2 }' 6>&
```

exec 6>&-

Vložený dokument

Velmi často může skript volat další skript nebo program, který vyžaduje nějaký vstup. Mechanis-mus vloženého dokumentu (takzvaný *here document*) představuje mechanismus, kterým shell čte
aktuální vstup až po řádek s nastaveným koncovým slovem. Všechny takto načtené řádky pak

budou použity jako standardní vstup pro příkaz.Díky tomuto mechanismu nemusíte data ukládat do samostatného souboru, můžete použít speci-ální znaky shellu a vypadá to mnohem lépe než hromada příkazů echo:

```
[jura@jv2 tldp]$ cat startsurf.sh #!/bin/bash

# Tento skript nabízí uživateli volbu mezi prohlížeči.

echo "V systému jsou nainstalovány následující prohlížeče:"

# začátek vloženého dokumentu cat << BROWSERS mozilla links lynx konqueror opera netscape BROWSERS # konec vloženého dokumentu

echo -n "Který chcete použít? " read browser

echo "Spouštím $browser, čekejte..." $browser &

[jura@jv2 tldp]$ ./startsurf.sh V systému jsou nainstalovány následující prohlížeče: mozilla links lynx konqueror opera netscape Který chcete použít? opera Spouštím opera, čekejte...
```

I když hovoříme o vloženém *dokumentu*, jde ve skutečnosti o konstrukci uvnitř skriptu. Následující příklad ukazuje, jak automaticky nainstalovat balíček, přestože je normálně požadováno potvrzení:

```
#!/bin/bash

# Tento skript pomocí yumu automaticky nainstaluje balíček.

if [ $# -lt 1 ]; then echo "Použití: $0 balíček." exit 1
fi

yum install $1 << CONFIRM y CONFIRM
```

A takto vypadá spuštěný skript. Jakmile se objeví výzva „Is this ok [y/N]“, skript si automaticky odpoví „y“:

```
[root@picon bin]# ./install.sh tuxracer Gathering header information file(s) from server(s) Server: Fedora Linux 2 - i386 - core Server: Fedora Linux 2 - i386 - freshrpms Server: JPackage 1.5 for Fedora Core 2 Server: JPackage 1.5, generic Server: Fedora Linux 2 - i386 - updates Finding updated packages Downloading needed headers Resolving dependencies Dependencies resolved I will do the following: [install: tuxracer 0.61-26.i386] Is this ok [y/N]: EnterDownloading Packages Running test transaction: Test transaction complete, Success! tuxracer 100 % done 1/1 Installed: tuxracer 0.61-26.i386 Transaction(s) Complete
```

Shrnutí

V této kapitole jsme si ukázali, jak uživatele informovat o průběhu skriptu a jak jej požádat
o vstup. Nejčastěji se k tomu používají příkazy echo a read. Hovořili jsme také o tom, jak lze pomocí deskriptorů a operátorů přesměrování používat ke vstupu a výstupu soubory a jak to lze kombinovat se vstupem zadávaným přímo uživatelem.

Zdůrazňujeme velký význam toho, aby skript uživateli poskytoval srozumitelné informace. Jakmi-le skript používá i někdo další, vždy je lepší poskytnout více informací než méně. Vložené doku-menty představují konstrukci pro vytváření seznamů uvnitř skriptů. Jejich prostřednictvím lze také bez uživatelské intervence provádět úkony, které jsou za normálních okolností interaktivní.

Cvičení

Následující cvičení představují praktické aplikace konstrukcí, o nichž jsme v této kapitole hovoři
li. Při práci nad skripty používejte testovací adresář, ve kterém není příliš mnoho souborů. Napište vždy část kódu, otestujte ji a teprve pak pokračujte dále, nesnažte se napsat všechno najednou.

Vytvořte skript, který se uživatele zeptá na jeho věk. Je-li věk větší nebo roven 18, řekněte uživateli, že už může pít alkohol. Je-li uživatel mladší, řekněte mu, za kolik let bude moci začít pít. U uživatelů starších 18 let navíc spočítejte, kolik už toho statisticky vzato vypil (100 litrů na osobu a rok).

Vytvořte skript, jehož jediným parametrem bude název souboru. Pomocí vloženého doku-mentu nabídněte uživateli několik způsobů komprimace souboru. Nabízené volby by měly zahrnovat gzip, bzip2, compress a zip.
Vytvořte skript homebackup, který automatizuje použití příkazu tar tak, že se vždy vytvo-ří záloha domovského adresáře s volbami cvp a záloha bude umístěna do /var/backups. Implementujte následující funkce:

Kontrola počtu parametrů. Skript nepoužívá žádné parametry, pokud uživatel nějaké zadá, vypište návod a ukončete skript.
Kontrola, zda je v cílovém adresáři dostatek místa pro uložení zálohy.
Zeptejte se uživatele, zda chce vytvořit úplnou nebo inkrementální zálohu. Pokud úplná záloha neexistuje nebo je starší než týden, neptejte se na nic a vytvořte rovnou úplnou zálohu.
Zálohu komprimujte vhodným program. Říkejte uživateli, co skript dělá, protože záloha a komprimace může nějakou dobu trvat.
Vypište hlášení o velikosti vytvořené komprimované zálohy.

Další potřebné informace naleznete pomocí info tar nebo v deváté kapitole příručky „Úvod do Linuxu“.

4. Vytvořte skript simple-useradd.sh pro přidání uživatele do systému. Skript by měl dělat následující věci:

Pracovat s jediným parametrem, jinak vypsat chybové hlášení.
Nalézt v souboru /etc/passwd první volné uživatelské ID a vypsat je.
Vytvořit pro uživatele privátní skupinu a vypsat ID této skupiny.
Zeptat se na následující údaje: komentář, výběr z možných shellů (kontrolovat správ-nost výběru), datum platnosti účtu, další skupiny, v nichž má být uživatel členem.
Po zjištění všech údajů zapsat potřebné řádky do souborů /etc/passwd, /etc/group a /etc/shadow; vytvořit domovský adresář uživatele se správnými přístupovými právy; přidat uživatele do sekundárních skupin.
Nastavit heslo na předdefinovaný text.

5. Přepište skript z kapitoly „Ověření existence souboru“ tak, aby si vstup vyžádal od uživa-tele a nečetl jej z příkazového řádku.

Opakované operace

V této kapitole budeme hovořit o následujících tématech:

Použití smyček for, while a until – jak zvolit vhodnou smyčku
Příkazy break a continue
Tvorba skriptů pomocí příkazu select
Tvorba skriptů s proměnným počtem parametrů

Smyčka for

Jak to funguje?

Smyčka for je první ze tří smyčkových konstrukcí shellu. Tato smyčka umožňuje zadat seznam hodnot. Následně se pro každou položku ze seznamu hodnot provede seznam příkazů. Syntaxe příkazu je následující:

for NÁZEV [in SEZNAM]; do PŘÍKAZY; done

Pokud není část [in SEZNAM] uvedena, je nahrazena "in $@", a příkazy se tedy provedou pro jed-notlivé poziční parametry (viz kapitolu „Speciální parametry“ a „Ověřování parametrů předáva-ných na příkazovém řádku“).

Návratový kód je roven návratovému kódu posledního provedeného příkazu. Pokud se SEZNAM neexpanduje na žádnou položku, a neprovede se tedy žádný příkaz, je návratový kód nula. Jako NÁZEV je možno zadat libovolnou proměnnou, velmi často se používá i. Jako SEZNAM je možno zadat libovolný seznam slov, řetězců nebo čísel, ať už literální nebo generovaný jiným pří-kazem. Jako prováděné PŘÍKAZY je možno zadat libovolné příkazy operačního systému, skripty, programy nebo příkazy shellu. Při prvním průchodu smyčkou obsahuje proměnná NAME první hodnotu ze SEZNAMu, při druhém průchodu druhou a tak dále. Smyčka končí ve chvíli, kdy se projde přes všechny hodnoty seznamu.

Příklady Specifikace seznamu položek pomocí substituce příkazu

První příklad je řádkový příkaz, který pomocí smyčky for vytvoří zálohu všech souborů .xml.

```
[carol@octarine ~/articles] ls *.xml file1.xml file2.xml file3.xml

[carol@octarine ~/articles] ls *.xml > list
[carol@octarine ~/articles] for i in `cat list`; do cp "$i" "$i".bak ; done

[carol@octarine ~/articles] ls *.xml*file1.xml file1.xml.bak file2.xml file2.xml.bak file3.xml file3.xml.bak
```

Další příklad vypisuje ty soubory z adresáře /sbin, které jsou pravděpodobně textové:

```
for i in `ls /sbin`; do file /sbin/$i | grep ASCII; done
```

Specifikace seznamu položek pomocí proměnné

Následující příklad ukazuje specifickou aplikaci, konvertující HTML soubory v určitém formátu na PHP soubory. Konverze probíhá tak, že odstraníme prvních 25 řádků a posledních 21 řádků, přičemž je nahradíme tagy jazyka PHP:

```
[carol@octarine ~/html] cat html2php.sh #!/bin/bash # konverze souborů html na php LIST="$(ls *.html)" for i in "$LIST"; do
        NEWNAME=$(ls "$i" | sed -e 's/html/php/')
        cat beginfile > "$NEWNAME"
        cat "$i" | sed -e '1,25d' | tac | sed -e '1,21d'| tac >> "$NEWNAME"
        cat endfile >> "$NEWNAME"

done
```

Protože neznáme počet řádků souboru, neumíme určit, od kterého řádku máme smazat oněch posledních 21. Problém řešíme příkazem tac, který vypisuje řádky souboru v obráceném pořadí.

Smyčka while

Co to je?

Konstrukce while slouží k opakovanému provádění seznamu příkazů tak dlouho, dokud je úspěš-ný řídící příkaz smyčky (jeho návratový kód je nula). Syntaxe smyčky je následující:

```
while ŘÍDÍCÍ_PŘÍKAZ; do PŘÍKAZY_CYKLU; done
```

Jako řídící příkaz je možno použít jakýkoliv příkaz či posloupnost příkazů, která vrací návratový kód. V rámci příkazů cyklu je možno použít libovolné programy, skripty nebo konstrukce shellu. Jakmile řídící příkaz proběhne neúspěšně, smyčka končí. Běh skriptu pak pokračuje prvním pří-kazem za done.

Návratový kód smyčky je roven návratovému kódu posledního provedeného příkazu těla, případně je nulový, pokud se tělo vůbec neprovádělo.

Příklady Jednoduchý příklad smyčky while

Krátký příklad pro netrpělivé:

```
#!/bin/bash

# Tento skript otevře 4 terminálová okna

i="0"

while [ $i -lt 4 ] do xterm & i=$[$i+1] done
```

Vnořené smyčky while

Následující skript je určen ke kopírování snímků z webové kamery na webový server. Každých pět minut je vytvořen obrázek, každou hodinu je vytvořen nový adresář pro obrázky z dané hodiny. Každý den se vytváří nový adresář se 24 podadresáři. Skript běží na pozadí.

```
#!/bin/bash


# Tento skript kopíruje soubory z domovského adresáře na# na webový server (pomocí scp a SSH klíčů)# Každou hodinu se vytváří nový adresář.

PICSDIR=/home/carol/pics
```

```
WEBDIR=/var/www/carol/webcam


while true; do DATE=`date +%Y%m%d` HOUR=`date +%H` mkdir $WEBDIR/"$DATE"

        while [ $HOUR -ne "00" ]; do
                DESTDIR=$WEBDIR/"$DATE"/"$HOUR"
                mkdir "$DESTDIR"
                mv $PICDIR/*.jpg "$DESTDIR"/
                sleep 3600
                HOUR=`date +%H`

        done done
```

Všimněte si příkazu true. Znamená to, že smyčka má pokračovat trvale, dokud nebude přeruše-na násilně (například příkazem kill nebo pomocí Ctrl+C).Následující krátký skript je možné použít k testování, každých pět minut generuje soubor:

```
#!/bin/bash

# Tento skript každých 5 minut generuje soubor

while true; do touch pic-`date +%s`.jpg sleep 300 done
```

Všimněte si použití příkazu date ke generování názvů souborů a adresářů. Více informací nalez-nete na jeho manuálové stránce.

Používejte služby systému

Výše uvedený příklad je pouze demonstrační. Pravidelné úkony je mnohem vhodnější pro-vádět prostřednictvím démona *cron*. Nezapomeňte na přesměrování výstupu a chybového výstupu skriptů, které tímto démonem spouštíte.

Řízení smyčky while z klávesnice

Následující skript je možné přerušit stiskem Ctrl+C:

```
#!/bin/bash

# Tento skript vám poradí

FORTUNE=/usr/games/fortune

while true; do echo "Na jaké téma chcete poradit?" cat << topics politics startrek kernelnewbies sports bofh-excuses magic love literature drugs education topics

echo echo -n "Vyberte si: " read topic echo echo "Rada zdarma na téma $topic: " echo $FORTUNE $topic echo

done
```

Všimněte si, že volby nabízíme podle vloženého dokumentu. Opět používáme podmínku true, která zajistí trvalé opakování smyčky až do jejího přerušení.

Výpočet průměru

Následující skript počítá průměr z uživatelem zadaných hodnot. Zadávané hodnoty se nejprve tes-tují, pokud jsou mimo povolený rozsah, vypíše se upozornění. Po zadání q smyčka skončí:

```
#!/bin/bash

# Výpočet průměru ze zadaných čísel.

SCORE="0" AVERAGE="0" SUM="0"
NUM="0"

while true; do

    echo -n "Zadejte skóre [0-100] ('q' ukončí program): "; read SCORE;
    if (("$SCORE" < "0") || ("$SCORE" > "100")); then
        echo "Ale no tak, zkuste to znovu: "

    elif [ "$SCORE" == "q" ]; then
```

```
        echo "Průměrné skóre: $AVERAGE%."
        break

    else
        SUM=$[$SUM + $SCORE]
        NUM=$[$NUM + 1]
        AVERAGE=$[$SUM / $NUM]

    fi

done

echo "Konec."
```

Všimněte si, že proměnné na posledních řádcích neuzavíráme do uvozovek, aby správně proběhly aritmetické výpočty.

Smyčka until

Co to je?

Smyčka until je velmi podobná smyčce while s tím rozdílem, že se její tělo provádí tak dlouho, dokud řídící příkaz neskončí úspěšně. Dokud je neúspěšný, smyčka se opakuje. Syntaxe je stejná jako u smyčky while:

until ŘÍDÍCÍ_PŘÍKAZ; do PŘÍKAZY_CYKLU; done

Návratový kód opět odpovídá návratovému kódu posledního příkazu těla cyklu anebo je nulový v případě, že se tělo vůbec neprovedlo. Řídícím příkazem může být opět cokoliv, co vrací návratový kód, tělo cyklu může být tvořeno libovolnými unixovými příkazy, skripty nebo konstrukcemi shellu.

Jak už víme, středník může být nahrazen jedním nebo více oddělovači řádku.

Příklad

Příklad ukazuje vylepšený skript picturesort.sh (viz kapitolu „Vnořené smyčky while“), který testuje velikost dostupného diskového prostoru. Pokud není k dispozici dostatek místa, odstraní snímky z předchozích měsíců:

```
#!/bin/bash


# Tento skript kopíruje soubory z domovského adresáře na# na webový server (pomocí scp a SSH klíčů)# Každou hodinu se vytváří nový adresář.# Není-li dostatek místa na disku, odstraní se nejstarší snímky.

        while true; do DISKFUL=$(df -h $WEBDIR | grep -v File | awk '{print $5 }' | cut -d "%"
                        -f1 -)

                until [ $DISKFUL -ge "90" ]; do
                                DATE=`date +%Y%m%d`
                                HOUR=`date +%H`
                                mkdir $WEBDIR/"$DATE"


                                while [ $HOUR -ne "00" ]; do DESTDIR=$WEBDIR/"$DATE"/"$HOUR" mkdir "$DESTDIR" mv $PICDIR/*.jpg
                                                "$DESTDIR"/ sleep 3600 HOUR=`date +%H`
                                done

                        DISKFULL=$(df -h $WEBDIR | grep -v File | awk '{ print $5 }' | cut -d
        "%" -f1 -) done

                        TOREMOVE=$(find $WEBDIR -type d -a -mtime +30)for i in
                        $TOREMOVE; dorm -rf "$i";done

done
```

Všimněte si inicializace proměnných HOUR a DISKFULL a použití voleb příkazů ls a date k získá-ní správného seznamu TOREMOVE odstraňovaných snímků.

Přesměrování vstupně-výstupních operací a smyčky

Přesměrování vstupu

Řízení smyčky nemusí být založeno na testování výsledku příkazu nebo na uživatelském vstupu. Je také možno načítat data ze souboru a řízení smyčky ponechat na výsledku čtení. V těchto přípa-dech se jako řídící příkaz velmi často používá příkaz read. Dokud do smyčky vstupují načítané řádky, pokračuje provádění těla smyčky. Jakmile dojde k přečtení celého vstupu, smyčka skončí.

Vzhledem k tomu, že je celá konstrukce smyčky chápána jako jediná příkazová struktura (napří-klad while ŘÍDÍCÍ_PŘÍKAZ; do TĚLO_SMYČKY; done), je nutné přesměrování provést až za příkazem done, takže bude splňovat základní tvar:

příkaz < soubor

Tento způsob přesměrování funguje i pro ostatní typy smyček.

Přesměrování výstupu

Následující příklad ukazuje použití příkazu find jako vstupu pro příkaz read, kterým je následně řízena smyčka while:

```
[carol@octarine ~/testdir] cat archiveoldstuff.sh#!/bin/bash

# Tento skript vytvoří v aktuálním adresáři podadresář, do nějž přesune# staré soubory.# Po úpravě lze volat z cronu a spouštět každý týden či měsíc.
ARCHIVENR=`date +%Y%m%d` DESTDIR="$PWD/archive-
$ARCHIVENR"

mkdir $DESTDIR

find $PWD -type f -a -mtime +5 | while read file do
  gzip "$file"; mv "$file".gz "$DESTDIR" echo "$file archivován" done
```

Před přesunutím do archivačního adresáře budou soubory komprimovány.

Break a continue

Vestavěný příkaz break

Příkaz break slouží k okamžitému ukončení aktuální smyčky bez ohledu na řídící podmínku. Pou-žívá se například v situacích, kdy nevíte, kolikrát smyčka proběhne, protože je závislá kupříkladu na uživatelském vstupu.

Následující příklad ukazuje uměle přerušovanou smyčku while. Jde o mírně vylepšenou verzi skriptu wisdom.sh z kapitoly „Řízení smyčky while z klávesnice“.

```
#!/bin/bash


# Tento skript vám poradí
# Nyní jej navíc lze korektně ukončit.


FORTUNE=/usr/games/fortune


while true; doecho "Na jaké téma chcete poradit?"echo "1. politics"echo "2.
startrek"echo "3. kernelnewbies"echo "4. sports"echo "5. bofh-excuses"echo "6.
magic"echo "7. love"echo "8. literature"echo "9. drugs"echo "10.
education"echo

echo -n "Zvolte číslo tématu, 0=konec: "
read choice
echo


case $choice in 1) $FORTUNE politics ;; 2) $FORTUNE startrek
        ;;
```

```
3)
$FORTUNE kernelnewbies
;;
4)
echo "Sport je ztráta času, energie a peněz."
echo "Vra se ke klávesnici."
echo -e "\t\t\t\t -- \"moje prostřední jméno je Nezdravý\""
;;
5)
$FORTUNE bofh-excuses
;;
6)
$FORTUNE magic
;;
7)
$FORTUNE love
;;
8)
$FORTUNE literature
;;
9)
$FORTUNE drugs
;;
10)
$FORTUNE education
;;
0)
echo "OK, čau!"
break
;;
*)
echo "Volba není platná, zvolte číslo 0 až 10."
;; esac done
```

Mějte na paměti, že příkaz break ukončí smyčku, nikoliv celý skript. Můžete si to vyzkoušet napří-klad tak, že na konec skriptu přidáte příkaz echo. Ten bude proveden poté, co byla (po zadání nuly) smyčka přerušena příkazem break.

Ve vnořených smyčkách je možné příkazu break říct, kolik vnořených úrovní má být přerušeno. Více informací naleznete na informačních stránkách bashe.

Vestavěný příkaz continue

Příkaz continue ihned pokračuje další iterací své smyčky for, while, until nebo select. Použijete-li jej ve smyčce for, řídící proměnná nabude následující hodnoty ze seznamu. Při pou-žití ve smyčkách while nebo until se ihned pokračuje vyhodnocením řídícího příkazu na začát-ku smyčky.

Příklady

V následujícím příkladu konvertujeme názvy souborů na malá písmena. Pokud není zapotřebí pro-vést konverzi názvu, pokračuje se v provádění smyčky příkazem continue. V tomto případě na systémových prostředcích nic moc neušetříme a stejného efektu bychom dosáhli i pomocí příka-zů sed či awk. Je ale vhodné tuto konstrukci znát zejména při provádění náročných úloh, kdy je možné testováním na vhodných místech smyčky ušetřit výkon.

```
[carol@octarine ~/test] cat tolower.sh#!/bin/bash

# Tento skript převede názvy všech souborů obsahujících velká písmena
# na názvy s malými písmeny.


LIST="$(ls)"


for name in "$LIST"; do


if [[ "$name" != *[[:upper:]]* ]]; then
continue
fi
```

```
ORIG="$name"
NEW=`echo $name | tr 'A-Z' 'a-z'`


mv "$ORIG" "$NEW"
echo "nový název souboru $ORIG je $NEW"
done
```

Skript má přinejmenším jednu nevýhodu – přepíše existující soubory. Volba noclobber v bashi by nám mohla pomoci, pouze pokud bychom používali přesměrování. Jistou ochranu nám poskyt-ne přepínač -b příkazu mv, ochrání nás však pouze před jedním přepsáním existujícího soubo-ru:

```
[carol@octarine ~/test] rm *

[carol@octarine ~/test] touch test Test TEST

[carol@octarine ~/test] bash -x tolower.sh ++ ls
+ LIST=test Test TEST
+ [[ test != *[[:upper:]]* ]]
+ continue
+ [[ Test != *[[:upper:]]* ]]
+ ORIG=Test ++ echo Test ++ tr A-Z a-z
+ NEW=test
+ mv -b Test test
+ echo 'nový název souboru Test je test' nový název souboru Test je test
+ [[ TEST != *[[:upper:]]* ]]
+ ORIG=TEST ++ echo TEST ++ tr A-Z a-z
+ NEW=test
+ mv -b TEST test
+ echo 'nový název souboru TEST je test' nový název souboru TEST je test

[carol@octarine ~/test] ls -a ./ ../ test test~
```

Příkaz tr je součástí balíčku *textutils*, zvládá všechny možné typy znakových transformací. Při pou-žití pozor na diakritiku, do množiny znaků a-z nepatří české znaky s diakritikou, musí se případ-ně ošetřit zvlášť.

Vytváření nabídek vestavěným příkazem select

Obecné Použití příkazu select

Příkaz select umožňuje snadné vytváření nabídek. Jeho syntaxe je podobná smyčce for:

select SLOVO [in SEZNAM]; do TĚLO_SMYČKY; done

Nejprve bude expandován SEZNAM a vznikne tak seznam položek. Výsledek expanze je vypsán nastandardní chybový výstup, každá položka je uvedena pořadovým číslem. Pokud není klauzule inSEZNAM uvedena, vypíší se poziční parametry skriptu, jako kdyby byl použit zápis in $@. Seznamse vypisuje pouze jednou.

Po vypsání všech položek se zobrazí prompt PS3 a načte se jeden řádek ze standardního vstupu.Pokud načtená hodnota odpovídá číslu některé z vypsaných položek, bude hodnota proměnnéSLOVO nastavena na název položky. Bude-li načtený řádek prázdný, znovu se zobrazí seznampoložek a prompt PS3. Je-li načten znak *EOF* (konec souboru), smyčka končí. Vzhledem k tomu,že většina uživatelů netuší, jak z klávesnice znak *EOF* zadat, je rozumné jako jednu z položeknabídnout volbu, která provede příkaz break. Zadání jakékoliv jiné hodnoty nastaví proměnnouSLOVO na prázdnou hodnotu.

Přečtený řádek je uložen do proměnné REPLY.Po každém načteném řádku se provede tělo smyčky, dokud ji neukončíte příkazem break.

Příklady

Následující příklad je velmi jednoduchý, není ale příliš uživatelsky přívětivý:

```
[carol@octarine testdir] cat private.sh#!/bin/bash

echo "Tento skript vám umožní nastavit kterýkoliv soubor v adresáři jako pri
```

```
vátní."
echo "Zvolte číslo souboru, který chcete chránit:"


select FILENAME in *;

do

        echo "Zvolili jste $FILENAME ($REPLY), nyní je přístupný pouze vám."
        chmod go-rwx "$FILENAME" done [carol@octarine
testdir] ./private.sh Tento skript vám umožní nastavit kterýkoliv soubor
v adresáři jako privátní. Zvolte číslo souboru, který chcete chránit: 1)
archive-20030129 2) bash 3) private.sh #? 1 Zvolili jste
archive-20030129 (1), nyní je přístupný pouze vám. #?
```

Skript vylepšíme nastavením promptu PS3 a nabídkou možnosti skript ukončit:

```
#!/bin/bash


echo "Tento skript vám umožní nastavit kterýkoliv soubor v adresáři jako pri
vátní."
echo "Zvolte číslo souboru, který chcete chránit:"


PS3="Vaše volba: "
QUIT="UKONČIT PROGRAM"
touch "$QUIT"


select FILENAME in *; do case
$FILENAME in
            "$QUIT")
                echo "Konec."
                break
                ;;

            *)
                echo "Zvolili jste $FILENAME ($REPLY)"
                chmod go-rwx "$FILENAME"
                ;;

  esac done rm "$QUIT"
```

Vnořené nabídky

Jako jeden z příkazů v těle konstrukce select můžete opět použít příkaz select a v rámci jedné volby tak nabídnout vnořené volby.Hodnota proměnné PS3 se při vstupu do vnořené smyčky nezmění. Pokud budete chtít vnořenounabídku odlišit jiným promptem, musíte nastavení proměnné na správných místech měnit.

Vestavěný příkaz shift

Co to dělá?

Příkaz shift je jedním z vestavěných příkazů Bourne shellu, který naleznete i v bashi. Příkaz pra-cuje s jediným parametrem, číslem. Zadané poziční parametry budou o toto číslo posunuty vlevo. Poziční parametry na pozicích $N+1 až $# tedy budou posunuty do proměnných $1 až $#-N+1.
Pokud byl tedy skript zavolán s deseti parametry a použijete příkaz shift 4, tak se z hodnoty $5 stane $1, ze $6 se stane $2 a tak dále. Ze $10 se stane $6 a původní hodnoty proměnných $1, $2 a $3 budou zahozeny.

Je-li N nula nebo větší než $# (celkový počet parametrů, viz kapitolu „Ověřování parametrů pře-dávaných na příkazovém řádku“), uspořádání parametrů se nezmění. Není-li N uvedeno, považu-je se za rovno jedné. Pokud bude N větší než $# nebo menší než nula, bude návratový kód nenu-lový, jinak bude roven nule.

Příklady

Příkaz shift se typicky používá v případech, kdy není počet parametrů dopředu znám, tedy napří-klad v situaci, kdy uživatel může zadat libovolný počet parametrů. V takových případech se para-metry typicky zpracovávají ve smyčce while s řídící podmínkou (($#)). Ta je pravdivá, dokud je počet parametrů nenulový. Jednotlivé parametry se pak zpracovávají v proměnné $1 s násled-ným příkazem shift. S každým voláním příkazu shift se počet parametrů snižuje, až nakonec dosáhne nuly a smyčka skončí.

Následující příklad, skript cleanup.sh, používá příkaz shift ke zpracování všech zadaných adre-sářů:

```
#!/bin/bash

# Tento skript maže soubory, které nebyly 365 dní použity.

USAGE="Použití: $0 adresář1 adresář2 adresář3 ... adresářN"

if [ "$#" == "0" ]; then
        echo "$USAGE"
        exit 1 fi

while (( "$#" )); do

    if [[ "$(ls $1)" == "" ]]; then echo "Prázdný adresář, nic se nemaže." else
            find $1 -type f -a -atime +365 -exec rm -i {} \; fi

shift

done
```

Exec vs. xargs

Výše použitý příkaz find je možno nahradit následujícím zápisem:

find volby | xargs [příkazy_k_provedení_nad_každým_nalezeným_souborem]

Příkaz xargs vytváří a spouští příkazové řádky načítané ze standardního vstupu. Výhodou je, že příkazový řádek je generován až do maximální délky povolené systémem. Až poté bude zavolán zadaný příkaz, v našem případě rm. Jsou-li na vstupu ještě další data, bude vytvořen další příkazový řádek. Při použití find -exec je příkaz volán pro každý nalezený soubor samostatně. Použitím příkazu xargs tedy dosáhnete podstatného zrychlení skriptu.

V následujícím příkladu modifikujeme skript z kapitoly „Vložený dokument“ tak, aby mohl najednou nainstalovat více balíčků:

```
#!/bin/bash
if [ $# -lt 1 ]; then echo "Použití: $0 balíček(čky)" exit 1
fi while (($#)); do
        yum install $1 << CONFIRM y CONFIRM shift done
```

Shrnutí

V této kapitole jsme si ukázali, jak je možné zajistit opakované provádění příkazů pomocí smy-ček. Nejčastěji se smyčky vytváří pomocí příkazů for, while a until, případně pomocí jejich kom-binací. Příkaz for provádí tělo smyčky předem daným počtem průchodů. Pokud nevíte, kolikrát má tělo proběhnout, použijte smyčky until nebo while.

Smyčku lze ukončit nebo přímo přeskočit na další iteraci pomocí příkazů break a continue. Jako vstup pro smyčku lze prostřednictvím operátoru přesměrování vstupu použít soubor, pomocí smyček je možné také číst výstup z příkazů, v takovém případě se výstup předává smyčce pomocí roury. Příkaz select se používá v interaktivních skriptech k vytváření nabídek. Zpracování řádkových parametrů v těle smyčky je možné provést pomocí příkazu shift.

Cvičení

Nezapomínejte: Když vytváříte skript, postupujte po částech a před začleněním do skriptu každou část otestujte.

Vytvořte skript, který pořídí (rekurzivní) kopii souborů v adresáři /etc, takže je budete moci bez obav editovat.
Napište skript, který bude pracovat s jediným parametrem, názvem adresáře. Pokud bude zadán jiný počet parametrů, vypište návod k použití skriptu. Není-li parametr názvem adre-sáře, vypište jiné chybové hlášení. Pro zadaný adresář vypište názvy pěti největších sou-borů a pěti naposledy změněných souborů.
Dokážete vysvětlit, proč bylo v příkladu z kapitoly „Přesměrování výstupu“ nutné uvést názvy proměnných v uvozovkách?

Vytvořte skript podobný tomu z kapitoly „Vestavěný příkaz break“ s tím, že po třech průchodech smyčkou bude skript ukončen.
Pro skript z kapitoly „Příklady“ vymyslete lepší ochranu před přepsáním souboru, než je přepínač -b příkazu mv. Můžete například testovat, jestli už soubor existuje. Nedělejte nic zbytečně!
Přepište skript whichdaemon.sh z kapitoly „Booleovské operace“ tak, aby:

vypsal seznam serverů, u nichž dokáže otestovat, zda jsou spuštěny (nabízí se napří-klad Apache, SSH server, NTP server, DNS server, správa napájení a podobně);
nechal uživatele zvolit server a následně vypsal rozumné údaje o konkrétní službě (například název webového serveru, stav NTP serveru a podobně);
umožnil uživateli testovat i jiné než skriptu známé servery, v těchto případech ověří pouze to, zda zadaný proces běží.

7. Podívejte se na skript v kapitole „Výpočet průměru“. Všimněte si, jak se zpracovává vstup jiných znaků než q, a ošetřete jej lépe.

Více o proměnných

V této kapitole budeme hovořit o pokročilejších metodách použití proměnných a parametrů. Po jejím prostudování byste měli zvládnout:

Deklarovat a používat pole
Specifikovat typ proměnné, kterou chcete použít
Nastavit proměnnou jen pro čtení
Nastavit hodnotu proměnné příkazem set

Typy proměnných

Obecné přiřazení hodnoty

Jak už jsme viděli, bash dokáže pracovat s různými druhy proměnných či parametrů. Do této chví-le jsme se příliš nestarali o to, jaký typ proměnných používáme, takže proměnným bylo možné přiřadit libovolnou hodnotu. Ukazuje to jednoduchý příklad:

[bob in ~] PROMENNA=*12*

[bob in ~] echo $PROMENNA 12

[bob in ~] PROMENNA=*řetezec*

[bob in ~] echo $PROMENNA řetězec

V některých případech je takové chování nežádoucí, například pokud pracujete s telefonními čísly či s jinými číselnými hodnotami. Kromě explicitního stanovení typu proměnné je občas vhodné deklarovat proměnnou jako konstantu. Často se to dělá na začátku skriptu. Od okamžiku dekla-race je možné už jen pracovat s hodnotou konstanty, hodnotu už není možno změnit. Proměnná může být také „seznamem“ více proměnných různých typů, pak hovoříme o takzvaném *poli* (VAR0, VAR1, ..., VARN).

Vestavěný příkaz declare

Pomocí příkazu declare můžete omezit možnosti přiřazení hodnoty proměnné. Syntaxe příkazu declare je následující:

declare VOLBA(Y) PROMĚNNÁ=hodnota

Následující tabulka uvádí seznam voleb, které je možné při deklaraci proměnných použít: Jestliže namísto symbolu „-“ použijete „+“, bude příslušný příznak dané proměnné zrušen. Použi

Volba	Význam
-a	Proměnná bude deklarována jako pole.
-f	Proměnná může obsahovat jen název funkce.
-i	Proměnná bude deklarována jako celočíselná, při přiřazování hodnoty proměnné se provede aritmetické

	vyhodnocení (viz kapitolu „Aritmetická expanze“).
-p	Zobrazí vlastnosti a hodnotu proměnných. Při použití volby -pse všechny ostatní volby ignorují.
-r	Proměnná bude deklarována jako konstanta. Proměnné nebude možno změnit hodnotu ani ji zrušit.
-t	Přiřadí proměnné příznak trace.
-x	Označí proměnnou jako určenou pro export prostředím.

Možné volby při deklaraci proměnných

jete-li příkaz declare ve funkci, vytvoří lokální proměnnou.Následující příklad ukazuje, jak stanovení typu ovlivní chování proměnné při přiřazování hodno-ty:

[bob in ~] declare -i PROMENNA=*12*

[bob in ~] PROMENNA=*řetězec*

[bob in ~] echo $PROMENNA 0

[bob in ~] declare -p PROMENNA declare -i PROMENNA="0"

Všimněte si, že bash umožňuje proměnnou explicitně deklarovat jako celočíselnou, neumožňuje však deklarovat proměnnou jako řetězec. Je to proto, že pokud není deklarováno jinak, proměnná může obsahovat libovolnou hodnotu.

[bob in ~] JINAPROMENNA=*cosi*

[bob in ~] declare -p JINAPROMENNA declare -- JINAPROMENNA="cosi"

Jakmile omezíte typ proměnné, bude moci obsahovat jen nastavený typ hodnot. Možná nastave
ní jsou celočíselné hodnoty, konstanty a pole.
Návratové kódy souvisejících operací naleznete na informačních stránkách bashe.

Konstanty

V bashi se konstanty vytvářejí tím, že se proměnná označí jako pouze pro čtení. Toto označení proměnné se provede vestavěným příkazem readonly. Jeho syntaxe je následující:

readonly VOLBY PROMĚNNÁ(é)

Hodnoty takto označených proměnných už nebude možno změnit dalšími příkazy přiřazení. Uve-dete-li volbu -f, uvedené proměnné se odkazují na funkce shellu, viz kapitolu „Funkce“. Uvede-te-li volbu -a, uvedené proměnné budou typu pole. Pokud neuvedete žádné parametry nebo pokud zadáte volbu -p, vypíše se seznam proměnných nastavených jen pro čtení. Při použití volby -p bude možné výstup příkazu přesměrovat na vstup dalšího příkazu.Návratový kód příkazu je nulový kromě situací, kdy byla zadána neplatná volba, některá z uve-dených proměnných či funkcí neexistuje nebo pokud použijete volbu -f na název proměnné,a nikoliv na název funkce.

[jura@jv2 ~]$ readonly TUX="síla tučňáka"

[jura@jv2 ~]$ TUX="majkro soft" bash: TUX: readonly variable

Pole

Vytváření polí

Pole je proměnná, která obsahuje více dalších proměnných. V rámci pole je možné používat různé typy proměnných. Velikost pole není omezena, není také nutné prvky pole indexovat nebo přiřazovat spojitě. Pole jsou indexována od nuly – první prvek má index 0.

Nepřímou deklaraci proměnné typu pole je možno provést následujícím příkazem přiřazení:

POLE[INDEX]=hodnota

Výraz *INDEX* je chápán jako aritmetický a výsledkem jeho vyhodnocení musí být nezáporné číslo. Explicitní deklarace pole se provede vestavěným příkazem declare:

declare -a POLE

Bude akceptována i deklarace s uvedením indexu, index se nicméně ignoruje. Pomocí příkazů declare a readonly je možné specifikovat příznaky pole, ty pak platí pro všechny prvky pole. Nelze používat pole se smíšenými příznaky prvků.

Pole je možno vytvořit také následujícím složeným přiřazením:

POLE=(hodnota1 hodnota2 ... hodnotaN)

Jednotlivé hodnoty je možno zapsat ve tvaru *[index=]řetězec*. Uvedení indexu je nepovinné. Pokud uvedete index, bude hodnota přiřazena prvku s tímto indexem, pokud jej neuvedete, bude hod-nota přiřazena prvku s indexem o jednu vyšším, než je poslední použitý index. Stejný formát zápi-su akceptuje i příkaz declare. Neuvedete-li žádný index, budou prvky pole indexovány od nuly.

Další prvky pole s libovolným indexem je možné vytvářet následujícím příkazem:

POLE[index]=hodnota

Vestavěný příkaz read podporuje volbu -a, která umožňuje číst nebo přiřazovat hodnoty prvkům pole.

Odkazování se na prvky pole

Chcete-li se odkazovat na prvky pole, použijte složené závorky. Tento zápis je nutný, aby se předešlo interpretaci expanzních operátorů. Pokud jako číslo indexu uvedete @ nebo *, odkážete se tím na všechny prvky pole.

```
[jura@jv2 ~]$ POLE=(raz dva tři)

[jura@jv2 ~]$ echo ${POLE[*]}
raz dva tři

[jura@jv2 ~]$ echo $POLE[*] raz[*]

[jura@jv2 ~]$ echo ${POLE[2]} tři

[jura@jv2 ~]$ POLE[3]=čtyři

[jura@jv2 ~]$ echo ${POLE[*]} raz dva tři čtyři
```

Budete-li se na pole odkazovat bez uvedení indexu, bude efekt stejný, jako kdybyste se odkázali na první prvek pole, tedy prvek s indexem 0.

Rušení prvků pole

Jednotlivé prvky pole, případně pole samotné, můžete zrušit vestavěným příkazem unset:

```
[jura@jv2 ~]$ unset POLE[1]

[jura@jv2 ~]$ echo ${POLE[*]} raz tři čtyři

[jura@jv2 ~]$ unset POLE

[jura@jv2 ~]$ echo ${POLE[*]} - nic -
```

Příklady

Praktické příklady na použití polí se hledají velmi těžce. Snadno najdete spoustu skriptů, kterépomocí polí řeší nějaké matematické výpočty, jinak ale nedělají nic pořádného. A to je pořád talepší situace, většina „příkladů“ totiž pouze v teoretické a překombinované rovině ukazuje, covšechno by šlo s poli udělat.

Je to dáno tím, že pole jsou už poměrně složité struktury. Záhy zjistíte, že věci, které by pomocípolí šly udělat, už jsou dávno implementovány – samozřejmě také s použitím polí, ale na nižšíúrovni, typicky v jazyce C, v němž je vytvořena většina unixových příkazů. Dobrým příkladem jepříkaz history. Pokud by vás to zajímalo, podívejte se do adresáře built-ins ve zdrojovém kódubashe, konkrétně na soubor fc.def.

Dalším důvodem, proč se vhodné příklady špatně hledají, je to, že ne všechny shelly pole pod

porují, takže jejich použití je vždy na úkor přenositelnosti.
Po delším hledání jsme našli následující příklad skriptu používaného jedním poskytovatelem.

Skript distribuuje konfigurační soubor webového serveru Apache na více strojů webové farmy:

```
#!/bin/bash

if [ $(whoami) != 'root' ]; then
        echo "$0 můžeš spustit jen jako root!"
        exit 1; fi if [ -z $1 ]; then
    echo "Použití: $0 </cesta/k/httpd.conf>" exit 1 fi

httpd_conf_new=$1 httpd_conf_path="/usr/local/apache/conf" login=htuser

farm_hosts=(web03 web04 web05 web06 web07)

for i in ${farm_hosts[@]}; do su $login -c "scp $httpd_conf_new ${i}:${httpd_conf_path}" su $login -c "ssh $i
        sudo /usr/local/apache/bin/apachectl graceful"

done exit 0
```

První dva testy pouze ověřují, zda skript spouští správný uživatel se správným parametrem. Názvy jednotlivých serverů ve farmě jsou deklarovány v poli farm_hosts. Na tyto stroje se vždy nako-píruje konfigurační soubor a provede se restart démona. Všimněte si, že se používají příkazy z balíku Secure Shell, které poskytují zabezpečenou komunikaci se servery. A také si všimněte, že i náš příklad je docela násilný a namísto použité konstrukce by stačilo použít jednodušší zápis:

for i in web03 web04 web05 web06 web07; do …

který by problém vyřešil stejně, ovšem s menší námahou. Pole by se nám vyplatilo až v případě opakovaného použití nebo při velkém množství serverů, kde by (možná) zajistilo trochu lepší přehlednost.

Další příklad pochází od Dana Richtera. Řešil následující problém:„...Na webových stránkách naší firmy zveřejňujeme různé demoverze a každý týden je jeden pra-covník povinen je všechny projít a vyzkoušet. Napsal jsem tedy skript, který má definováno pole
testerů, příkazem date +%W zjistí číslo týdne a operací modulo nalezne správný index do pole.Vybranému „šťastlivci“ pak pošle mail.“Řešení vypadá takto:

```
#!/bin/bash
# get-tester-address.sh
#
# Nejprve ověříme, zda bash podporuje pole.
# (Podpora polí byla přidána poměrně nedávno.)
#
whotest[0]='test' || (echo 'Chyba: tato verze bashe nepodporuje pole' && exit 2)


#
# Seznam testerů.
#
wholist=(

        'Bob Smith <bob@example.com>"Jane L. Williams <jane@example.com>"Eric S. Raymond <esr@example.com>"Larry Wall
        <wall@example.com>"Linus Torvalds <linus@example.com>'
    ) # # Zjistíme počet testerů.[1] # (Smyčka, dokud nenačteme prázdný řetězec.) # count=0 while [ "x${wholist[count]}" != "x" ] do
    count=$(( $count + 1 )) done

## A te zjistíme, kdo je na řadě.#week=`date '+%W'` # Číslo týdne (0..53).week=${week#0} # Zrušíme případnou uvozující nulu.

let "index = $week % $count" # číslo_týdne modulo počet_testerů =
šastný_výherce


email=${wholist[index]} # zjistíme mail "výherce"
```

```
echo $email # a vypíšeme jej
```

Tento skript se pak používá z jiných skriptů, například z tohoto:

```
email=`get-tester-address.sh` # Zjistíme, kdo je tento týden na řadě. hostname=`hostname` # Lokální počítač.

# # A pošleme mail tomu, kdo si to zaslouží. # mail $email -s '[Testovani demoverzi]' <<EOF Šastným testerem pro tento týden je: $email

Připomenutí - věci k otestování najdete tady: http://web.example.com:8080/DemoSites

(Tuto zprávu vygeneroval $0 na ${hostname}.) EOF
```

Operace s proměnnými

Aritmetické operace

O této problematice jsme podrobně hovořili v kapitole „Aritmetická expanze“.

Délka proměnné

Pomocí zápisu ${#PROMĚNNÁ} je možné zjistit délku hodnoty proměnné ve znacích. Je-li proměn-ná polem, bude vrácen počet prvků pole. Ukazuje to následující příklad:

```
[jura@jv2 ~]$ echo $SHELL /bin/bash
```

[1] Pozn. překl.: Autor skriptu zřejmě nedával pozor, když se probírala konstrukce ${#wholist}.

```
[jura@jv2 ~]$ echo ${#SHELL} 9

[jura@jv2 ~]$ POLE=(raz dva tři)

[jura@jv2 ~]$ echo ${#POLE} 3
```

Operace je „UNICODE safe“:

```
[jura@jv2 ~]$ COSI=ěščř [jura@jv2 ~]$ echo ${#COSI} 4
```

Transformace proměnných Substituce

${PROMĚNNÁ:-*HODNOTA*}

Pokud je PROMĚNNÁ nedefinována nebo je prázdná, bude výsledkem výrazu *HODNOTA*, jinak jím bude hodnota PROMĚNNÉ:

```
[bob in ~] echo ${TEST:-test} test

[bob in ~] echo $TEST


[bob in ~] export TEST=cosi

[bob in ~] echo ${TEST:-test} cosi

[bob in ~] echo ${TEST2:-$TEST} cosi
```

Tento obrat se velmi často používá v podmínkách, například:

```
[ -z "${COLUMNS:-}" ] && COLUMNS=80
```

je jen kratší zápis pro

```
if [ -z "${COLUMNS:-}" ]; then
        COLUMNS=80 fi
```

Podrobnější informace o tomto typu podmínek naleznete v kapitole „Porovnávání řetězců“. Pokud místo pomlčky (-) uvedete rovnítko (=), bude příslušná hodnota rovnou přiřazena neexistující či prázdné proměnné:

```
[bob in ~] echo $TEST2


[bob in ~] echo ${TEST2:=$TEST} cosi
```

[bob in ~] echo $TEST2 cosi

Následující konstrukci můžete použít k otestování existence proměnné. Pokud proměnná není nastavena, bude na standardní výstup vypsán alternativní text a neinteraktivní shell bude ukončen:

```
[jura@jv2 tldp]$ cat testvar.sh #!/bin/bash

# Tento skript testuje, zda je proměnná nastavena. Pokud není, # skončí s chybovým hlášením.

echo ${TESTVAR:?"Chci znát hodnotu proměnné TESTVAR..."} echo "Proměnná TESTVAR je nastavena,
můžeme pokračovat."

[jura@jv2 tldp]$ ./testvar.sh./testvar.sh: line 6: TESTVAR: Chci znát hodnotu proměnné TESTVAR...

[jura@jv2 tldp]$ export TESTVAR="máme nastaveno"

[jura@jv2 tldp]$ ./testvar.sh máme nastaveno Proměnná TESTVAR je nastavena, můžeme pokračovat.
```

Získání části řetězce

Pokud z řetězcové proměnné potřebujete ponechat jen vybraný podřetězec, použijte následující syntaxi:

${PROMĚNNÁ:*OFFSET*:*DÉLKA*}

Výsledkem je DÉLKA znaků počínaje znakem OFFSET, offset se počítá od nuly. Neuvedete-li délku, budou vráceny všechny znaky až do konce řetězce.

```
[jura@jv2 tldp]$ export RETEZEC="nejakydlouhytext"

[jura@jv2 tldp]$ echo ${RETEZEC:4} kydlouhytext

[jura@jv2 tldp]$ echo ${RETEZEC:6:5} dlouh
```

Vynechání části řetězce

Konstrukce

${PROMĚNNÁ#*SLOVO*}

a

${PROMĚNNÁ##*SLOVO*}

slouží k vymazání expanze *SLOVA* z proměnné PROMĚNNÁ. Nejprve dojde k expanzi *SLOVA* a vytvo-ření vzoru, stejně jako by šlo o expanzi názvů souborů. Pokud vzniklý vzor odpovídá začátku expandované hodnoty PROMĚNNÉ, výsledkem celé expanze bude expandovaná hodnota PROMĚNNÉ s užitím nejkratšího vyhovujícího vzoru („#“) nebo nejdelšího vyhovujícího vzoru („##“).

Bude-li hodnota PROMĚNNÉ* nebo @, operace vynechání vzoru bude uplatněna na všechny pozič-ní parametry a výsledkem expanze bude takto vzniklý seznam.
Bude-li PROMĚNNÁ pole s uvedením indexu „*“ nebo „@“, bude vynechání provedeno nad všemi prvky pole. Ukazuje to následující příklad:

```
[jura@jv2 ~]$ echo ${POLE[*]} dva pět devět pět

[jura@jv2 ~]$ echo ${POLE[*]#pět} dva devět

[jura@jv2 ~]$ echo ${POLE[*]#d} va pět evět pět

[jura@jv2 ~]$ echo ${POLE[*]#d*} va pět evět pět

[jura@jv2 ~]$ echo ${POLE[*]##d*} pět pět
```

Opačného efektu dosáhneme pomocí operátorů „%“ a „%%“, které odstraňují znaky od konce řetězce tak, jak to ukazuje následující příklad:

```
[jura@jv2 ~]$ echo $JMENO totojestrašlivědlouhéjméno

[jura@jv2 ~]$ echo ${JMENO%jméno} totojestrašlivědlouhé
```

Nahrazení části řetězce

Tuto operaci je možné provést zápisem:

${PROMĚNNÁ/*CO_NAHRADIT*/*ČÍM_NAHRADIT*}

nebo:

${PROMĚNNÁ//*CO_NAHRADIT*/*ČÍM_NAHRADIT*}

V prvním případě bude nahrazen pouze první výskyt hledaného řetězce, ve druhém případě budou nahrazeny všechny výskyty:

[jura@jv2 ~]$ echo ${JMENO/strašlivě/hrozně} totojehroznědlouhéjméno

Další podrobnosti naleznete na informačních stránkách bashe.

Shrnutí

Za normálních okolností může proměnná obsahovat jakýkoliv datový typ, toto chování lze změ-nit explicitní deklarací proměnné. Vestavěným příkazem readonly je možné deklarovat kon-stanty.

Pole obsahuje množinu proměnných. Je-li deklarován datový typ, musí být všechny prvky pole daného typu.Bash umožňuje snadno substituovat a transformovat hodnoty proměnných. Mezi standardní ope-race patří zjištění délky proměnné, aritmetické operace s proměnnými a substituce části řetězce.

Cvičení

Několik náročnějších úloh:

1. Vytvořte skript, který provede následující činnosti:

Zobrazí název spuštěného skriptu.
Zobrazí první, třetí a desátý parametr skriptu.
Zobrazí celkový počet parametrů předaných skriptu.
Jsou-li předány více než tři parametry, příkazem shift posune všechny parametry o tři pozice vlevo.
Následně vypíše zbývající parametry.
A vypíše počet parametrů.Vyzkoušejte skript s žádným, jedním, třemi a více než deseti parametry.

2. Napište skript, který bude implementovat jednoduchý textový webový prohlížeč pomocí příkazů wget a links -dump. Uživatel má tři možnosti: zadat URL, zadat b a vrátit se tak k předchozí stránce anebo zadat q a ukončit tak prohlížeč. V poli se ukládá posledních 10 zadaných URL, na něž se uživatel může vracet.

Funkce

V této kapitole budeme hovořit o následujících tématech:

Co to jsou funkce
Vytváření a výpis funkcí z příkazového řádku
Funkce ve skriptech
Předávání parametrů funkcím
Kdy funkce použít

Úvod

Co to jsou funkce?

Funkce představují mechanismus, jak seskupit posloupnost příkazů a následně je spustit pouhým zadáním názvu funkce. Název funkce musí být v rámci shellu či v rámci skriptu jedinečný. Všech-ny příkazy, které tvoří tělo funkce, se provádějí úplně normálně. Jakmile funkci zavoláte uvede-ním jejího názvu, provede se seznam příkazů, asociovaný s danou funkcí. Funkce se

provádí ve stejném shellu, v jakém byla deklarována, k její interpretaci se nespouští nová instance shellu.

Při vyhodnocování názvu zadaného příkazu se před názvy funkcí rozpoznávají jen vybrané vesta-věné příkazy shellu, a to konkrétně break, :, ., continue, eval, exec, exit, export, readonly, return, set, shift, trap a unset.

Syntaxe

Funkce se deklarují jednou z následujících dvou syntaxí:

function FUNKCE { PŘÍKAZY; }

nebo

FUNKCE () { PŘÍKAZY; }

Oba zápisy shodně deklarují funkci s názvem FUNKCE. Uvedení klíčového slova function není povinné, v takovém případě je ale nutné za názvem funkce uvést závorky.
Ve složených závorkách je deklarováno tělo funkce. Příkazy těla se provedou vždy, když jako název příkazu zadáte název deklarované funkce. Návratový kód funkce je roven návratovému kódu posledního provedeného příkazu těla.

> Běžné chyby
>
> Mezi složenými závorkami a tělem funkce musí být mezery, jinak bude zápis interpretován nesprávně.
>
> Tělo funkce musí končit středníkem nebo novým řádkem.

Poziční parametry ve funkcích

Funkce jsou jako miniaturní skripty – mohou zpracovávat parametry, mohou pracovat s proměn-nými viditelnými jen v těle funkce (pomocí vestavěného příkazu local) a mohou volajícímu skrip-tu vracet hodnoty.

Funkce používají vlastní mechanismus interpretace pozičních parametrů. Poziční parametry funk
ce se ovšem liší od pozičních parametrů volajícího skriptu. Parametry předávané funkci při jejím volání se uvnitř funkce chovají jako poziční parametry. Spe-ciální parametr # odpovídá počtu parametrů předaných funkci. Hodnota parametru 0 se nemění a odpovídá názvu běžícího skriptu. Pomocí proměnné FUNCNAME je možné v těle funkce zjistit její název.

Použijete-li v těle funkce příkaz return, provádění funkce se ukončí a pokračuje se příkazem následujícím za voláním funkce. Po skončení provádění funkce budou hodnoty pozičních para-metrů a parametru # obnoveny na původní hodnoty. Je-li příkaz return volán s číselným para-metrem, bude tato hodnota vrácena jako návratový kód funkce. Jednoduchý příklad:

```
[jura@jv2 tldp]$ cat showparams.sh #!/bin/bash

echo "Tento skript demonstruje použití parametrů funkce." echo

echo "Poziční parametr 1 tohoto skriptu má hodnotu $1." echo

test ()
{
echo "Poziční parametr 1 funkce je $1."
RETURN_VALUE=$?
echo "Návratový kód funkce je $RETURN_VALUE."

}

test jiný_parametr

[jura@jv2 tldp]$ ./showparams.sh nějaký_parametr Tento skript demonstruje použití parametrů
funkce.

Poziční parametr 1 tohoto skriptu má hodnotu nějaký_parametr.

Poziční parametr 1 funkce je jiný_parametr. Návratový kód funkce je 0.
```

Návratová hodnota funkce se velmi často ukládá do proměnné, aby s ní bylo možno později pra-covat. Inicializační skripty systému velmi často ukládají návratovou hodnotu do proměnné RETVAL a následně ji různě testují:

```
                                        if [ $RETVAL -eq 0 ]; then
        <spuštění démona>
```

Nebo jiný příklad ze skriptu /etc/init.d/amd, ve kterém se používá optimalizačních mechanis-mů bashe:

```
[ $RETVAL = 0 ] && touch /var/lock/subsys/amd
```

Příkazy za operátorem && se provedou pouze v případě, že předchozí podmínka byla splněna.
Jde o zkrácenou reprezentaci konstrukce if/then/fi.Návratový kód funkce často poslouží jako návratový kód celého skriptu. Řada inicializačníchskriptů končí něčím jako exit $RETVAL.

Vypsání funkce

Všechny funkce známé aktuálnímu shellu je možné vypsat vestavěným příkazem set bez para-metrů. Funkci je možné volat opakovaně, dokud ji nezrušíte příkazem unset. Funkce vypisuje i příkaz which:

```
[lydia@cointreau ~] which zless zless is a function zless () {
        zcat "$@" | "$PAGER" }

[lydia@cointreau ~] echo $PAGER less
```

Tyto druhy funkcí se typicky deklarují v uživatelských inicializačních skriptech shellu. Funkce jsou univerzálnější než aliasy a nabízejí snadný a jednoduchý způsob úprav uživatelského prostředí. Jedna funkce pro uživatele DOSu:

```
dir () {
        ls -F --color=auto -lF --color=always "$@" | less -r }
```

Příklady funkcí ve skriptech

Recyklace

V systému naleznete celou řadu skriptů, které využívají funkcí jakožto strukturovaného mecha-nismu pro provedení posloupnosti příkazů. Na některých distribucích naleznete například soubor /etc/rc.d/init.d/functions, který je používán všemi inicializačními skripty. Díky tomu je možné běžné úkony, jako je kontrola, zda běží proces, spuštění démona, ukončení démona a podobně, naprogramovat pouze jednou v obecné podobě. Na všech místech, kde se tyto úkony provádějí, se pak „recykluje“ tento univerzální kód. Následující funkce checkpid pochází právě ze souboru functions:

```
# Kontrola zda běží úloha s daným(i) PID checkpid() {
        local i for i in $* ; do
                [ -d "/proc/$i" ] && return 0
        done
        return 1 }
```

Tuto funkci používá několik dalších funkcí ve stejném skriptu, ty jsou následně volány z dalších skriptů. Například funkci daemon používá většina inicializačních skriptů, které spouštějí nějaký server.

Nastavení cesty

Následující kód naleznete v souboru /etc/profile. Deklaruje se funkce pathmunge, která se následně používá k nastavení cesty pro uživatele *root* a pro ostatní uživatele:

```
        pathmunge () { if ! echo $PATH | /bin/egrep -q
                "(^|:)$1($|:)" ; then if [ "$2" = "after" ] ;
                then PATH=$PATH:$1 else PATH=
                $1:$PATH fi fi }

# Nastavení cesty
if [ `id -u` = 0 ]; then
        pathmunge /sbin
        pathmunge /usr/sbin
        pathmunge /usr/local/sbin fi

pathmunge /usr/X11R6/bin after

unset pathmunge
```

První parametr funkce je název adresáře. Pokud adresář v aktuální cestě není, přidá se do ní.

Druhý parametr určuje, zda má být adresář přidán na začátek nebo na konec stávající cesty. Normálním uživatelům se do cesty přidává jen adresář /usr/X11R6/bin, uživateli *root* se přidává několik dalších adresářů obsahujících systémové příkazy. Na konci skriptu se funkce ruší, takže později už neexistuje.

Vzdálené zálohování

Následující příklad používá autor k zálohování textu knih. Pomocí SSH klíčů se skript přihlašuje na vzdálený server. Jsou definovány dvě funkce, buplinux a bupbash, které vytvoří soubor .tar, zkomprimují jej a odešlou na vzdálený server. Lokální kopie se následně vymaže.

V neděli se spouští pouze funkce bupbash.

```
#/bin/bash

LOGFILE="/nethome/tille/log/backupscript.log" echo "Začíná zálohování dne `date`" >>
"$LOGFILE"

buplinux()
{
DIR="/nethome/tille/xml/db/linux-basics/"
TAR="Linux.tar"
BZIP="$TAR.bz2"
SERVER="rincewind"
RDIR="/var/www/intra/tille/html/training/"


cd "$DIR"
tar cf "$TAR" src/*.xml src/images/*.png src/images/*.eps
echo "Komprimuji $TAR..." >> "$LOGFILE"
bzip2 "$TAR"
echo "...hotovo." >> "$LOGFILE"
echo "Kopíruji na $SERVER..." >> "$LOGFILE"
scp "$BZIP" "$SERVER:$RDIR" > /dev/null 2>&1
echo "...hotovo." >> "$LOGFILE"
echo -e "Záloha příručky Linux hotova:\nZálohovány byly zdrojové soubory
a obrázky PNG a EPS.\nPomocné soubory odstraněny." >> "$LOGFILE"
rm "$BZIP"

}

bupbash()
{
DIR="/nethome/tille/xml/db/"
TAR="Bash.tar"
BZIP="$TAR.bz2"
FILES="bash-programming/"
SERVER="rincewind"
RDIR="/var/www/intra/tille/html/training/"


cd "$DIR"
tar cf "$TAR" "$FILES"
echo "Komprimuji $TAR..." >> "$LOGFILE"
bzip2 "$TAR"
echo "...hotovo." >> "$LOGFILE"
echo "Kopíruji na $SERVER..." >> "$LOGFILE"
scp "$BZIP" "$SERVER:$RDIR" > /dev/null 2>&1
echo "...hotovo." >> "$LOGFILE"


echo -e "Záloha příručky Bash hotova:\nZálohovány byly soubory $FILES\nPomocné
soubory odstraněny." >> "$LOGFILE"
rm "$BZIP"

}
```

```
DAY=`date +%w`

if [ "$DAY" -lt "2" ]; then echo "Je `date +%A`, zálohuje se jen Bash." >> "$LOGFILE" bupbash
else buplinux bupbash
fi


echo -e "Záloha pro den `date` HOTOVA\n----------" >> "$LOGFILE"
```

Skript se spouští z cronu, tedy bez interakce s uživatelem. Proto přesměrováváme standardní chy-bový výstup příkazu scp do /dev/null.Můžete namítnout, že jednotlivé kroky bylo možno sloučit do jediného příkazu, například:

tar c dir_to_backup/ | bzip2 | ssh server "cat > backup.tar.bz2"

Pokud vás ovšem zajímají výsledky jednotlivých operací, což může být užitečné v případě selhá-ní skriptu, není takové spojení žádoucí. Zápis

příkaz &> soubor

je ekvivalentní zápisu

příkaz > soubor 2>&1

Shrnutí

Funkce představují jednoduchý způsob, jak seskupit více příkazů, které chcete provést opakova-ně. V těle funkce jsou poziční parametry skriptu nahrazeny parametry funkce. Jakmile funkce skončí, poziční parametry jsou obnoveny na původní hodnotu. Funkce jsou jako miniaturní skrip-ty a stejně jako skripty mohou vracet návratový kód.

I když byla tato kapitola poměrně krátká, její znalost je nezbytná pro dosažení profesionální míry lenosti, což je standardní cíl všech systémových administrátorů.

Příklady

Toto jsou dvě užitečné věci, které můžete vyřešit pomocí funkcí:

1. Přidejte do konfiguračního souboru ~/.bashrc funkci, která automatizuje tisk manuálo-vých stránek. Výsledkem by mělo být, že zadáte něco jako printman <příkaz> a přísluš-ná manuálová stránka vyjede z tiskárny. Skript můžete otestovat pomocí zařízení pseudo-tiskárny.
Jako bonus můžete funkci rozšířit o možnost volby sekce manuálu, v němž má být stránka vyhledána.
Ve svém domovském adresáři si vytvořte podadresář, ve kterém budete ukládat definice funkcí. Uložte do něj několik funkcí. Užitečné funkce mohou být například takové, které emulují vybrané příkazy DOSu či komerčních UNIXů. Při zpracovávání souboru ~/.bas-hrc budou funkce importovány do vašeho prostředí.

Zpracování signálů

V této kapitole budeme hovořit o následujících tématech:

Existující signály
Použití signálů
Použití příkazu trap
Jak uživateli zabránit v přerušení programu

Signály

Úvod Manuálové stránky signálů

Seznam všech dostupných signálů je popsán na manuálových stránkách, v závislosti na operač-ním systému ale může jít o různé stránky. Na většině linuxových systémů jde o stránku man 7 signal. V případě pochybností můžete stránku najít například takto:

man -k signal | grep list

nebo:

apropos signal | grep list

Názvy signálů zjistíte také příkazem kill -l.

Signály pro bash

Pokud nejsou nastaveny trapy, interaktivní bash ignoruje signály *SIGTERM* a *SIGQUIT*. Signál *SIGINT* je zachycován a zpracováván. Pokud je zapnuto řízení úloh, ignorují se také signály *SIGTTIN*, *SIGTTOU* a *SIGTSTP*. Jsou-li tyto signály generovány z klávesnice, ignorují je i příkazy spouštěné z bashe mechanismem substituce příkazů.

Signál *SIGHUP* standardně ukončí shell. Interaktivní shell pošle signál *SIGHUP* všem úlohám, běží-cím i pozastaveným. Pokud byste chtěli toto chování pro určitý proces změnit, podívejte se do dokumentace na vestavěný příkaz disown. Pokud budete chtít po přijetí signálu *SIGHUP* zabít všechny úlohy, pomocí vestavěného příkazu shopt nastavte volbu huponexit.

Posílání signálů pomocí shellu

Prostřednictvím bashe je možné poslat následující signály: Nastavení terminálu Zkontrolujte nastavení příkazu stty. Používáte-li „moderní“ emulátory terminálu, je poza-stavení a obnovení výstupu obvykle vypnuto. Standardní xterm ve výchozím nastavení podporuje kombinace Ctrl+S a Ctrl+Q.

Klávesová kombinace	Význam
	Signál přerušení, úloze na popředí se pošle SIGINT.
	Odložené pozastavení. Běžící proces se zastaví, jakmile se pokusí číst z terminálu. Řízení se vrací shellu, uživatel může proces poslat na popředí, na pozadí nebo jej ukončit. Odložené pozastavení je podporováno jen v některých systémech.
	Pozastavení. Běžícímu procesu se pošle signál SIGTSTP, proces se zastaví a řízení se vrací shellu.

Řídící signály v bashi

Příkaz kill

Většina moderních shellů včetně bashe obsahuje vestavěnou funkci kill. V bashi je možno zadatjako volbu název nebo číslo signálu, parametrem je ID úlohy či procesu. Volbou -l je možnovypsat návratový kód – je nulový, pokud se podařilo úspěšně odeslat alespoň jeden signál, a nenulový, pokud došlo k chybě.

Příkaz kill z adresáře /usr/bin může nabízet i další možnosti, například zabití procesu patřícíhojinému uživateli nebo zadání procesu názvem podobně jako v příkazech pgrep a pkill.Není-li specifikován signál, oba příkazy kill posílají signál *SIGTERM*.Nejběžnější signály shrnuje následující tabulka:

Název signálu	Hodnota signálu	Efekt
SIGHUP	1	Zavěšení (reinicializace).
SIGINT	2	Přerušení z klávesnice.
SIGKILL	9	Zabití.
SIGTERM	15	Ukončení.
SIGSTOP	17, 19, 23	Zastavení.

Běžné signály

SIGKILL a SIGSTOP

Signály *SIGKILL* a *SIGSTOP* nelze zachytit, blokovat ani ignorovat.

Chcete-li ukončit proces nebo více procesů, obvyklý postup je začít nejméně nebezpečným sig-nálem, *SIGTERM*. Programy, které se starají o své korektní ukončení, tak mají šanci provést ukon-čovací proceduru iniciovanou právě signálem *SIGTERM* a například smazat dočasné soubory nebo uzavřít otevřené soubory. Pokud procesu pošlete rovnou signál *SIGKILL*, nebude mít možnost pro-vést své korektní ukončení, což může vést k nežádoucím efektům.

Samozřejmě pokud nefunguje korektní ukončení, jedinou šancí jsou signály *SIGINT* nebo *SIGKILL*. Pokud proces nereaguje na Ctrl+C, můžete jej ukončit příkazem kill -9:

maud: ~> ps -ef | grep *stuck_process* maud 5607 2214 0 20:05 pts/5 00:00:02 stuck_process maud: ~> kill -9 *5607*

maud: ~> ps -ef | grep *stuck_process* maud 5614 2214 0 20:15 pts/5 00:00:00 grep stuck_process [1]+ Killed stuck_process

Jestliže proces běží ve více instancích, je jednodušší použít příkaz killall. Používá stejné volby jako příkaz kill, provede se však pro všechny instance daného procesu. Pokud byste tento příkaz chtěli používat v produkčním prostředí, nejprve si ověřte jeho chování, protože na některých komerčních Unixech se nemusí chovat očekávaným způsobem.

Trapy

Obecné

V některých situacích může být nežádoucí, aby měl uživatel možnost skript ukončit klávesovou kombinací, například protože se čeká na zadání nějakého vstupu nebo je nutné provést úklid. Tyto sekvence je možné zachytit příkazem trap a po jejich zachycení je možno provést zadanou sekvenci příkazů.

Syntaxe příkazu trap je velmi prostá:

trap [PŘÍKAZY] [SIGNÁLY]

Tímto příkazem nastavíte zachycování uvedených signálů, přičemž je možné signály zadávat buď názvem (s nebo bez prefixu *SIG*) nebo číslem. Pokud nastavíte zachycování signálu *0* či *EXIT*, pří-kazy se provedou při skončení shellu. Pokud nastavíte zachycování signálu *DEBUG*, bude se nastavená sekvence příkazů provádět po provedení každého příkazu skriptu. Je také možné defi-novat signál *ERR*, v takovém případě se sekvence příkazů provede pokaždé, když některý z pří-kazů skriptu skončí s nenulovým návratovým kódem. Sekvence se neprovede v případě, pocháze-li nenulový návratový kód z příkazu if, while nebo until. Neprovádí se ani v případech, je-li nenulový návratový kód výsledkem logických operátorů *AND* (&&) nebo *OR* (||) nebo pokud byl návratový kód příkazu invertován operátorem negace (!).

Návratový kód samotného příkazu trap je nulový, ledaže byste zadali neplatný seznam signálů.
Příkaz trap používá několik voleb, které jsou popsány na informačních stránkách bashe. Následující velmi jednoduchý příklad zachycuje stisk kláves Ctrl+C a vypisuje hlášení. Pokud se pokusíte skript ukončit jinak než signálem *KILL*, nic se nestane:

```
#!/bin/bash # traptest.sh

trap "echo Bum!" SIGINT SIGTERM

echo "pid je $$"

while :                 # totéž jako "while true"
do
        sleep 60        # skript nic užitečného nedělá
done
```

Jak bash interpretuje trapy

Pokud bash zachytí signál, pro nějž je nastaven trap, a čeká se na dokončení příkazu, bude trap spuštěn až poté, co příkaz skončí. Jestliže prostřednictvím vestavěného příkazu wait bash čeká na asynchronní příkaz, způsobí přijetí signálu s nastaveným trapem okamžité ukončení příkazu wait s návratovým kódem větším než 128, ihned poté se provede trap.

Další příklady Detekce použití proměnné

Při ladění delších skriptů může být užitečné nastavit některým proměnným atribut *trace* a pro danou proměnnou zachycovat signál *DEBUG*. Za normálních okolností se proměnné deklarují pří-kazem přiřazení ve tvaru PROMĚNNÁ=hodnota. Pokud použijete následující deklaraci, můžete se dozvědět podrobnější informace o tom, co skript právě dělá:

```
declare -t PROMĚNNÁ=hodnota

trap "echo Právě byla použita PROMĚNNÁ." DEBUG

# zbytek skriptu
```

Závěrečný úklid

Příkaz whatis využívá databázi, která se pravidelně vytváří skriptem makewhatis.cron spouště-ným z cronu:

```
#!/bin/bash

LOCKFILE=/var/lock/makewhatis.lock

# Předchozí makewhatis musel úspěšně skončit:

[ -f $LOCKFILE ] && exit 0

# Nakonec smažeme zámek.

trap "{ rm -f $LOCKFILE ; exit 255; }" EXIT

touch $LOCKFILE makewhatis -u -w exit 0
```

Shrnutí

Signály je možné programům posílat příkazem kill nebo klávesovými zkratkami. Pomocí příkazu trap je možné signály zachycovat a reagovat na ně.Některé programy mohou signály ignorovat. Jediný signál, který žádný program ignorovat nemů-že, je signál *SIGKILL*.

Cvičení

Dvojice praktických příkladů:

Vytvořte skript, který pomocí příkazu dd zapíše na disketu bootovací obraz systému. Pokud se uživatel pokusí přerušit skript stiskem Ctrl+C, vypište hlášení, že vytvořená dis-keta nebude použitelná, a ukončete skript.
Vytvořte skript, který bude automatizovat instalaci balíčků. Balíček se stáhne z Internetu, dekomprimuje, rozbalí a přeloží. Vlastní instalace balíčku musí být nepřerušitelná.

ČÁST IV

Linux na cestách

Úvod

Mobilní počítačová zařízení (laptopy, notebooky, PDA, mobilní telefony, přenosné přehrávače hudby a videa, digitální fotoaparáty, kalkulačky a podobně) se od klasických stolních počítačů liší. Používají specifické hardwarové komponenty, například karty PCMCIA, infračervené a bluetooth porty, bezdrátová síťová rozhraní, LCD displeje, baterie, dokovací stanice. Jednotlivé hardwarové komponenty nelze měnit tak snadno, jako například grafickou kartu ve stolním počítači. Hardwa-rové prostředky (například diskový prostor nebo výkon CPU) bývají často limitovány. I přesto se výkonnostní rozdíl oproti stolním počítačům snižuje, takže v řadě případů může notebook sloužit jako plnohodnotná náhrada stolního počítače.

Podpora mobilních počítačových zařízení (například grafických čipů nebo interních modemů) bývá v Linuxu (a jiných alternativních operačních systémech) omezenější. Tato zařízení často pou-žívají specializovaný hardware, takže nalezení vhodného ovladače může být složitější. Navíc se tato zařízení často provozují v různých prostředích, takže je nutná podpora více konfigurací a s tím související bezpečnostní strategie.

Tato příručka představuje stručný souhrn informací týkajících se mobilních počítačových zařízení, které dnes můžete nalézt v celé řadě dokumentů HOWTO týkajících se laptopů, notebooků, PDA a mobilních telefonů. Popisujeme také různé specificky linuxové operace, jako například způso-by instalace laptopů, notebooků a PDA nebo různé konfigurace pro různá (síťová) prostředí.

I přes některá úskalí představuje Linux pro přenosná počítačová zařízení lepší volbu než většina ostatních operačních systémů, protože podporuje celou řadu způsobů instalace, pracuje v mnoha heterogenních prostředích a může mít menší nároky na hardwarové prostředky počítače.

Jaký notebook koupit?

Úvod

Přenosné počítače je možno rozdělit do různých kategorií. Jde o subjektivní rozdělení, přesto se o něj ale pokusíme. Naše rozdělení zhruba odpovídá běžně uváděným marketingovým kategoriím. Kritéria pro rozhodování jsou následující:

1. Hmotnost – často bývá skryta v označeních jako přenosný počítač, laptop/notebook, sub/mininotebook, palmtop nebo PDA. Neexistuje standardní způsob specifikace hmot-nosti notebooku, proto je nutné údaje uváděné výrobci (a dále citované v této příručce) považovat pouze za orientační. Otázkou totiž je, nakolik jsou v deklarované hmotnosti zahrnuty i napájecí zdroj (ať už externí nebo interní) nebo výměnné komponenty (jako dis-ketová či CD mechanika).
Různé periferie mohou být neuvěřitelně těžké. Může se stát, že subnotebook s různými exter-ními mechanikami, kabely, expandéry portů a napájecími zdroji bude v konečném důsledku těžší než zařízení, které má všechny potřebné věci integrovány. Subnotebooky jsou typicky užitečné pouze tehdy, pokud jste schopni je využít bez všech ostatních příslušenství.
Podporované operační systémy – proprietární versus otevřené.
Cena – neznačkové versus značkové.
Hardwarové parametry – velikost displeje, kapacita disku, rychlost procesoru, typ baterií a podobně.
Podpora Linuxu – grafický čip, zvuková karta, infračervený ovladač (IrDA), interní modem a podobně.

Přenosné počítače, laptopy/notebooky, sub/mini-notebooky, palmtopy, PDA/HPC

Přenosné počítače

Hmotnost větší než 4 kg. Stejné vlastnosti jako PC, pouze v menším provedení a s LCD displejem. S postupujícím vývojem a miniaturizací těchto modelů spíše ubývá, přesto má téměř každý výrob-ce notebooků v sortimentu nějakého zástupce tohoto typu.

Laptopy, notebooky

Hmotnost mezi 1,7 a 4 kg. Typicky obsahuje vlastní hardware a obvykle speciální procesor vyba-vený šetřícími režimy pro úsporu energie. Příklady: HP OmniBook 3100, Compaq Armada 1592DT, Thinkpad T43, Apple MacBook Pro. Termíny *laptop* a *notebook* považujeme za synonyma.

Subnotebooky, mininotebooky

Hmotnost mezi 1,3 a 1,7 kg. Typicky zahrnuje externí disketové a CD mechaniky. Příklady: HP OmniBook 800CT, Toshiba Libretto 100, Compaq Aero, Sony VAIO 505 nebo Thinkpad X41.

Palmtopy

Hmotnost mezi 0,7 a 1,3 kg. Typicky jsou vybaveny proprietárním komerčním operačním systémem. Příklady: HP 200LX.

PDA (Personal Digital Assistant) a HPC (Handheld PC)

Hmotnost pod 0,7 kg. Typicky jsou vybaveny proprietárním komerčním operačním systémem a velmi často jiným než Intel procesorem se systémy jako PalmOS, EPOC32, GEOS, Windows CE. Příklady: Newton Message Pad, Palm III (dříve Pilot), Psion série 3 a 5, Casio Z-7000.

Wearables

Hodinky, digitální pera, kalkulačky, digitální fotoaparáty, mobilní telefony a další.

Funkčnost v Linuxu

V důsledku nedostatečné podpory některých výrobců nemusí být všechny funkce notebooků vždy plně podporovány nebo plně funkční. Typicky problémovými komponentami jsou grafické čipy, IrDA porty, zvukové karty, řadiče PCMCIA, PnP zařízení a interní modemy. Před koupí noteboo-ku si o těchto tématech zjistěte co nejvíce informací. Ne vždy je ovšem snadné takovéto informa-ce získat. V některých případech nenaleznete potřebné údaje ani ve specifikaci, a dokonce je ne-sděluje ani výrobce. V kapitole „Podrobnosti o mobilním hardwaru“ proto uvádíme údaje o kom-patibilitě s Linuxem.

Mohli by vás také zajímat výrobci, kteří dodávají notebooky přímo s Linuxem. Objednáte-li si note-book s předinstalovaným Linuxem, můžete si ušetřit spoustu hádání a instalace doplňujících balíčků. Informace o těchto výrobcích můžete najít na serveru TuxMobil, *http://tuxmobil.org/reseller.html.*

V ČR jsou s Linuxem dodávány například notebooky Thinkpad (*http://www.thinkpad.cz*), se kte-rými lze objednat SUSE enterprise desktop, nebo notebooky Barbone, které jsou již dlouhou dobu dodávány s Mandriva Linuxem (*http://www.barbone.cz*). S toutéž distribucí se dodávají i notebo-oky certifikované Intelem (*http://www.vbi.cz*). Prozkoumejte též nabídku lokálních dodavatelů, u kterých je Linux čím dál tím více zastoupen. Téměř libovolnou distribuci vám nainstalují firmy, které se Linuxem přímo zabývají, viz *http://instalace.linux.cz/.*

Hlavní hardwarové vlastnosti

Kromě funkčnosti v Linuxu je nutné při výběru notebooku zvážit některé z hlavních hardwaro-vých vlastností těchto zařízení. Informace o funkčnosti v Linuxu naleznete v kapitole „Podrobnosti
o mobilním hardwaru“.

Hmotnost

Nepodceňujte hmotnost notebooku. Bývá ovlivněna zejména následujícími faktory:

velikost obrazovky,
typ baterií,
interní komponenty jako CD mechanika nebo disketová mechanika,
napájecí zdroj,
materiál, z nějž je zařízení vyrobeno.

Displej

Moderní notebooky obsahují *aktivní* maticové displeje (TFT). Laptopy s *pasivními* maticovými dis-pleji (DSTN) se už nevyrábějí. Aktivní displeje mají lepší barvy a kontrast, jsou však dražší a mají větší spotřebu. Zvažte také velikost displeje. Notebooky se běžně vyrábějí s displeji s úhlopříčkou až 17 palců. Větší displej víc váží, víc stojí a je méně skladný, je však lepší náhradou stolního počítače.

Baterie

Mezi obvykle používané typy baterií dnes patří *lithiumiontové* (*LiIon*), *niklmetalhydridové* (*NiMH*)
nebo *niklkadmiové* (*NiCd*). Většina současných notebooků obsahuje lithiumiontové baterie. LiIon baterie jsou nejdražší, při stejné kapacitě jsou ale mnohem lehčí než NiCd baterie a mají minimální paměťový efekt. NiMH baterie jsou lepší než NiCd, stále jsou ale poměrně těžké a trpí paměťovým efektem, i když menším než NiCd baterie.

Většina notebooků je bohužel vybavena bateriemi v proprietárním mechanickém provedení, takže baterie nelze mezi různými modely zaměňovat.

CPU Podporované rodiny CPU

Informace o tom, které hardwarové platformy linuxové jádro podporuje, naleznete ve FAQ kon-ference LKLM, *http://www.tux.org/lkml/.*

i286 – Tuto procesorovou rodinu Linux doposud nepodporuje, byť se o implementaci této podpory snaží projekt ELKS,

http://elks.sourceforge.net/. Pokud chcete, můžete namísto Linuxu použít Minix (*http://www.minix3.org/*), což je rovněž zdarma dostupný operační systém unixového typu. Minix podporuje platformy 8088 až 286 s pouhými 640 KB pamě-ti. Na adrese *http://tuxmobil.org/286_mobile.html* můžete zjistit, že se přímo prodávají notebooky se systémy ELKS i Minix.
i386 – Tato rodina zahrnuje celou skupinu procesorů kompatibilních s platformou Intel, patří sem Intel 386, 486, Pentium, Pentium Pro a Pentium II a kompatibilní procesory výrobců jako AMD, Cyrix a dalších. Většina současných notebooků používá procesory právě z této rodiny a jejich podpora Linuxem je tak bezproblémová.
m68k – Sem patří Amiga a Atari s procesory Motorola 680x0 (x>=2) s MMU a starší mode-ly Apple/Macintosh. Na čipu m68k byla založena dlouhá série notebooků Apple Powe-rBook a dalších – Macintosh Portable (první škaredý osmikilový pokus), PowerBook 100, 140, 170, 145, 160, 180c, 165c, 520c, 540c, 550c, 190; Duo 210, 230, 250, 270c, 280. Note-booky PowerBook Duo se vyráběly ve stejné době jako PowerBooky, byly to subnotebo-oky navržené tak, že je bylo možno zapojit do základové stanice (DuoDock), která měla více paměti, periferií a podobně, takže šly použít jako stolní počítače. Prvními PowerBoo-ky s PowerPC byly PowerBook 5300 (následoval po 190) a Duo 2300c. Úplný seznam všech vyráběných modelů Macintosh i s jejich specifikacemi naleznete na stránkách *http://www.apple-history.com/.* Výsledky instalace Linuxu na této platformě naleznete na stránkách *http://tuxmobil.org/apple.html.* Upozorňujeme, že na noteboocích 68k nelze pro-vozovat LinuxPPC, přestože to název naznačuje. LinuxPPC je určen pouze pro procesory

> PowerPC. Informace o použití Linuxu na platformě m68k naleznete na stránkách *http://www.mac.linux-m68k.org/.* „Podobně jako u notebooků na platformě Intel mají i notebooky Mac obecně různé obtížně zjistitelné hardwarové konfigurace. Navíc je těch-to notebooků k testovacím účelům k dispozici poměrně málo. Máme proto informace
> o nabootování Linuxu jen na strojích Powerbook 145, Powerbook 150, Powerbook 170, Powerbook 180 a Powerbook 190. I v případě úspěšného nabootování ovšem není k dis-pozici ADB powerbookového stylu, nefunguje APM a v zásadě téměř nic jiného. Jediná možnost přihlášení je tak terminál připojený k sériovému portu – to máme pozitivně vyzkoušeno na modelu 170."
>
> „Některé Powerbooky mají podporované interní IDE, ovladače pro PCMCIA mohou vznik-nout, pouze pokud někdo dá k dispozici potřebné informace. Je nutná také podpora FPU. Většina pozdějších modelů používá procesor 68LC040 bez FPU, řada procesorů má nefunkč-ní mechanismus FPU trapů a nelze tak na nich spustit linuxové binárky ani v režimu emula-ce FPU. Aktuální stav podpory modelů Powerbook 140, 160, 165, 165c, 180c, 190, 520 a Duo 210, 230, 250, 270c, 280 a 280c není znám." Existují i dva notebooky Atari, k nimž také nemáme dostatek informací. Následující citace pocházejí ze stránek Atari Gallery (*http://yescrew.atari.org/eng/atari.htm*). „Model STacy byl uveden krátce po Mega ST jako přenosný počítač Atari. Modely STacy se dodávaly se systémem TOS v1.04. Model ST Book byl zamýšlen jako náhrada STacy, který byl víceméně jen přenosná podoba modelu ST. ST Book představuje stejný výpočetní výkon jako ST, ovšem v podobě lehkého notebooku. Tento model byl prodáván pouze v Evropě a jen v omezeném množství. ST Book byl dodá-ván s TOS v2.06." Leon Stok, *stok@yis.nl,* říká: „Na modelech STacy a ST Book nelze provo-zovat Linux, protože jsou založeny na procesoru 68000, který nemá jednotku MMU. Pokud víme, Amiga nikdy notebooky nevyráběla. Jedna společnost nabízela sadu pro předělání stol-ní Amigy na přenosnou. Vycházeli ze standardních základních desek Amiga, takže na těch-to strojích lze použít jakoukoliv variantu Linuxu, která běží na standardních Amigách."

PowerPC (PPC) – I když některé linuxové ovladače dostupné pro platformy Intel nejsou pro PPC k dispozici, představuje Linux PPC plně použitelný systém pro platformu Macintosh PowerBook. Další informace můžete najít na stránkách *http://penguinppc.org.* Mimochodem, týmu kolem projektu iMac Linux (*http://www.imaclinux.net/*) se podařilo nabootovat Linux na iMAC DV. Na stránkách projektu naleznete také informace o Linuxu na iBooku.
Alpha, Sparc, Sparc64 – Podpora pro tyto architektury je teprve v začátcích. Pokud víme, notebooků s procesory Sparc a Alpha existuje jen několik, například Tadpole (*http://www.tadpole.com/*) a NatureTech (*http://www.naturetech.com.tw/*). Server TuxMobil se věnuje i tomuto tématu, *http://tuxmobil.org/mobile_solaris.html.*
StrongARM – Procesory s velmi nízkou spotřebou, aktivně podporované projektem Debian, které najdete i v některých zařízeních s WinCE, například HP Jornadas. V portaci Linuxu na tyto malé modely s dlouhou výdrží baterií brání pouze nedostatek technické specifikace. Plnohodnotný notebook založený na StrongARM by byl výtečnou linuxovou platformou. O PDA s procesory ARM/StrongARM hovoříme v kapitole „Personal Digital Assistant (PDA)".
MIPS – Tento procesor se používá v mainframech SGI a v serverech Cobalt Micro, čipy vycházející z této architektury se používají i v mnoha systémech s WindowsCE. Na někte-ré z nich byl portován i Linux.
Procesory AMD – Více informací o Linuxu a procesorech AMD naleznete na stránkách *http://www.x86-64.org* a *http://tuxmobil.org/cpu_amd.html.*
64bitové procesory – O této problematice rovněž hovoří server TuxMobil, *http:// tuxmobil.org/cpu_64bit.html.* Podpora 64bitových procesorů od Intelu i AMD se rapidně

> zlepšuje zároveň s jejich stoupající oblibou – tuto platformu dnes podporují všechny rozšířené distribuce.

Různé

Při vyšších rychlostech mají procesory větší spotřebu a produkují více tepla. V řadě notebooků se proto používají speciální procesory se sníženou spotřebou. Tyto procesory mají typicky nižší výkon než procesory ve stolních počítačích a jsou také dražší. Proto také u notebooků se standardními stolními procesory narazíte na mnohem hlučnější větrání.

Počet točivých zařízení

U laptopů a notebooků se často uvádí počet točivých zařízení.

Jedno: pevný disk. Typicky v subnoteboocích s možností připojení externí CD/DVD mechaniky.
Dvě: pevný disk a optická mechanika (CD/DVD).
Tři: pevný disk, optická mechanika (CD/DVD) a disketová mechanika (dnes již ne tak častá). Tyto notebooky často slouží jako náhrada stolních počítačů.

Chlazení

Velmi důležitý bod. Cokoliv založeného na procesorech PPC nebo Pentium (a výše) generuje velké množství tepla, které je nutno odvést. Obecně k tomu slouží buď ventilátor nebo (pasivní) chladič tvořený skříní počítače. Pokud je notebook chlazen ventilátorem, nesmí být tok vzduchu blokován, jinak by mohlo dojít k přehřátí. Počítače s větráním zespodu jsou proto velmi nešikovné – nelze je použít na měkkém podkladu.

Kvalita klávesnice

I když delší texty budete psát na stolním počítači, kvalitní klávesnice na notebooku vám ušetří spoustu trápení a bolení rukou. Všímejte si zejména rozložení speciálních kláves jako Esc, Tab, End, PageUp a PageDown a kurzorových kláves.

Cena

V porovnání se stolními počítači jsou notebooky poměrně drahé (i když při zahrnutí věcí jako LCD, IrDA a PCMCIA není cenový rozdíl až tak značný). Můžete se rozhodovat mezi značkou a neznačkovými modely. Doporučujeme vám sice volbu *neznačkového* modelu, má to ale svá úskalí. Notebooky často přicházejí k úhoně, je proto lepší, máte-li k dispozici kvalitní servis – a ten bývá většinou k dispozici jen pro značkové výrobky. Můžete se dokonce rozhodnout pro koupi notebooku z druhé ruky. Trh s notebooky se velmi často mění. Nová generace notebooků (ve smyslu nových procesorů, kapacit disků, rozměrů displeje a podobně) se objevuje přibližně každé tři měsíce, takže notebooky velmi rychle zastarávají. Toto stárnutí se ovšem neprojevuje na cenách použitých modelů, které tak vycházejí poměrně draze. Pokud byste se nicméně pro koupi použi-tého modelu rozhodli, dodržujte naše rady týkající se kontroly počítače.

Napájecí zdroj

Pokud cestujete do zahraničí, ověřte si, jaké napájecí napětí zvládne váš zdroj. Ten také bývá jednou z nejtěžších částí notebooku. Další zrada může být tvar zástrčky, který se stát od státu liší.

Další zdroje informací

Specifikace, manuály a podpora výrobců velmi často nestačí. Doporučujeme vám proto i další zdroje informací:

Server TuxMobil, který na adrese *http://tuxmobil.org/mylaptops.html* hovoří o noteboocích, Linuxu a dalších Unixech (například BSD).
Server Linux on Laptops, *http://www.linux-on-laptops.com/*.

Obecné informace o podpoře různých výrobců naleznete na adrese *http://tuxmobil.org/laptop_ manufacturer.html,* nečekejte však, že by výrobci Linux nějak přehnaně podporovali. Užitečný je také odkaz *http://tuxmobil.org/laptop_oem.html,* který vám může pomoci najít váš notebook pod jiným označením.

Kompatibilita s Linuxem

Související dokumentace

Hardware HOWTO, *http://tldp.org/HOWTO/Hardware-HOWTO/.*
Kernel HOWTO, *http://tldp.org/HOWTO/Kernel-HOWTO/.*
PCMCIA HOWTO, *http://pcmcia-cs.sourceforge.net/ftp/doc/PCMCIA-HOWTO.html.*
PCI HOWTO, *http://tldp.org/HOWTO/PCI-HOWTO.html.*
Plug-and-Play HOWTO, *http://tldp.org/HOWTO/Plug-and-Play-HOWTO.html.*

Obecné metody kontroly

Pokud na výše uvedených odkazech nezjistíte potřebné informace, je pátrání na vás. Linux vám naštěstí nabízí mnoho pomůcek. Podrobnější informace naleznete v kapitole „Podrobnosti
o mobilním hardwaru“. Obecně můžete použít:

Informace samotného jádra. Hledejte, jaký hardware dokázalo jádro detekovat. Tyto infor-mace uvidíte při bootování systému, můžete je vypsat příkazem dmesg a naleznete je také v souboru /var/log/messages. Prvotní hlášení bootovacího procesu naleznete v soubo-ru /var/log/boot.
Pokud používáte jádro s podporou souborového systému /proc, zjistíte podrobné infor-mace o PCI zařízeních příkazem cat /proc/pci. Přečtěte si soubor pci.txt, který je sou-částí dokumentace jádra. Další informace o neznámých PCI zařízeních nabízí projekt PCI ID Repository, *http://pciids.sourceforge.net/.* Můžete též použít příkaz lspci z balíku pci-utils.
Informace o zařízeních Plug-and-Play (PnP) zjistíte příkazem isapnp-tools (platí pouze pro starší zařízení na sběrnici ISA).
Informace o zařízeních SCSI zjistíte příkazem scsiinfo nebo programem scsi_info Davida Hindse.

Pokud kvůli detekci hardwaru nechcete instalovat nějakou kompletní linuxovou distribuci, může-te použít nějakou mikrodistribuci, viz kapitolu „Štíhlé aplikace a distribuce“. Balík muLinux obsa-huje dokonce program systest, TomsRtBt nabízí program memtest. Chcete-li memtest použít, zkopírujte jej příkazem dd if=/usr/lib/memtest of=/dev/fd0 na disketu a nabootujte systém z této diskety.
Pokud máte na notebooku nainstalovány Windows, můžete celou řadu informací o hardwaru zjis
tit v nich. Nabootujte do DOSu nebo do Windows, kde tyto informace snadno naleznete. Chcete-li údaje o hardwaru zjistit ve Windows9x/NT, spusťte Start -> Nastavení -> Ovládací pane-ly -> Systém -> Správce zařízení a všechno si opište. V MS-DOS a Windows 3.1x můžete použít příkaz msd, MicroSoft Diagnostics. K dispozici je také celá řada sharewarových nástrojů jako check-it, dr.hard a další. V novějších verzích Windows hledejte v Ovládacích panelech položku Systém a v ní záložku Hardware.

V některých případech je obtížné zjistit i to, kdo vlastně zařízení nebo některou jeho část vyrobil. Jestliže zařízení bylo certifikováno pro americký trh, naleznete na něm přidělené FCC ID a na stránkách Federální komise pro komunikace, FCC, můžete výrobce nalézt v databázi schválených zařízení. Databázi naleznete na adrese *http://www.fcc.gov/oet/fccid/help.html.*

Řada notebooků má problémy s kompatibilitou jak v Linuxu, tak i ve Windows. David Hinds, autor ovladačů PCMCIA, uvádí, že notebooky Toshiba používají proprietární PCMCIA bridge, který vykazuje stejné chybné chování ve Windows i v Linuxu. IBM Thinkpad má vážné problémy s BIOSem, které ovlivňují doručování událostí APM démonovi. Tyto chyby se projevují i ve Win-dows, v dokumentaci IBM jsou označeny jako „ke zvážení“.

Některé nekompatibility mohou být jen dočasné – pokud například notebook obsahuje USB řadič od Intelu, lze se v brzké době nadít plné podpory USB.

Vytvoření ovladače zařízení

Pokud narazíte na zařízení, které v Linuxu není doposud podporováno, nezapomínejte, že máte možnost napsat si ovladač sami. Doporučujeme vám knihu Linux Device Drivers Alessandra Rubi-niho a Andy Orama. Na adrese *http://www.oreilly.com/openbook/* je k dispozici její volně dostup-ná verze.

Koupě notebooku z druhé ruky

Několik doporučení, co před koupí notebooku z druhé ruky zkontrolovat:

Zkontrolujte povrch skříně, zda není viditelně poškozen.
Zkontrolujte chybné body na displeji, můžete si pomoci zvětšovacím sklem. Mimochodem, přípustnou chybovost displeje a další věci upravuje norma ISO 13406-2.
Zkuste vstupně-výstupní stress-test, například nástrojem bonnie.
Otestujte paměť nástroji jako memtest a crashme.
Zkuste stress-test procesoru, například příkazem md5sum/dev/urandom nebo překla-dem jádra.
Vyzkoušejte naformátovat disketu v disketové mechanice.
Vyzkoušejte čtení a zápis v CD/DVD mechanice.
Testování baterií je trochu složitější, protože si vyžádá více času – jeden nabíjecí a jeden pracovní cyklus. Můžete si pomoci nástrojem jako battery-stats, ten však nabízí pouze podporu APM, doposud není podporováno ACPI.
Povrch disku můžete zkontrolovat nástrojem e2fsck. Dalšími linuxovými nástroji jsou dosfsck a ostatní fsck.
Nedestruktivní test celého disku, zjištění jeho výkonu a velikosti provedete příkazem time dd if=/dev/hda of=/dev/null bs=1024k.
Vyzkoušet můžete také specializovanou distribuci Stresslinux (*http://www.stresslinux.org/*), která je určena pro zátěžové testy hardwaru a obsahuje mnohé z výše uvedených nástrojů.
Ověřte, zda počítač není kradený. Odkazy na databáze kradených notebooků naleznete například na adrese *http://tuxmobil.org/stolen_laptops.html.*

Pokud je nám známo, neexistuje linuxový nástroj srovnatelný s komplexními dosovými nástroji typu Check-It, Dr. Hard, Sysdiag a podobné. Tyto nástroje integrují mnoho různých testů. Podle našeho názoru je nejlepším nástrojem PC Diagnostics 95 Cragia Harta, naleznete jej na adrese *http://members.datafast.net.au/~dft0802/.* Navzdory číslu 95 v názvu jde o čistě dosový nástroj, spolehlivý, zdarma dostupný a malý (program má 76 KB, data 199 KB). Bohužel neumí otestovat IrDA port.

Dovolujeme si citovat část z dokumentace k tomuto programu: „Program je určen pro použití zku-šenými techniky. NENÍ určen k

používání ignoranty, kteří o fungování počítače nic nevědí. Výpi-sy nejsou pěkné, jsou ale funkční. Zjištěné hodnoty nejsou vysvětleny, protože cílem není nikoho vzdělávat. Tento program by měl být vnímán podobně jako jakýkoliv jiný kus nářadí. Aby bylo jeho užití k něčemu, je nutné jej používat správně a ve vhodné situaci. Pokud jste koncový uži-vatel, který se v počítačovém hardwaru nevyzná, není pro vás program vhodný.“

Notebooky se na rozdíl od stolních počítačů skutečně opotřebovávají. Lithiové baterie mají životnost až tisíc nabíjecích cyklů, ale mnohdy i méně. Klávesnice se vychodí, podsvícení displeje vybledne, tlačítka myši se ošoupají. Nejhorší je opotřebení konektorů v důsledku vibrací, které vede k občasným selháním (například při stisknutí klávesy Enter). Donesla se k nám historka o zničení notebooku za jednu cestu vlakem.

Bez hardwarových doporučení

Obecně je velmi těžké doporučovat jakékoliv modely notebooků. Svou roli hrají konkrétní potře-by uživatele a trh se vyvíjí velmi rychle. Zhruba každé tři měsíce přichází na trh nová generace notebooků (s většími disky, rychlejšími procesory a větším displejem). Proto se nebudeme pouš-tět do doporučování konkrétních modelů nebo značek. Doporučujeme vám ale na adrese *http://tuxmobil.org/laptop_manufacturer.html* ověřit, jak jsou na tom jednotliví výrobci s podpo-rou Linuxu.

Dobrý způsob na ověření hardwarové kompatibility s Linuxem představuje CD či DVD se systé-mem Knoppix. Detekce hardwaru v tomto systému pracuje velmi kvalitně a často dokáže dete-kovat veškeré komponenty notebooku. Podobnou službu zastanou i live-CD nebo live-DVD verze distribucí openSUSE, Mandriva nebo Fedora.

Dodavatelé notebooků a PDA kompatibilních s Linuxem

Vhodného prodejce můžete najít například na adrese *http://tuxmobil.org/reseller.html.* Někteří pro-dávají i notebooky bez předinstalovaného operačního systému od Microsoftu.Získat notebook bez předinstalovaného operačního systému od Microsoftu může být docela těžké.Pokud vás to zajímá, na adrese *http://tuxmobil.org/ms_tax.html* se můžete dočíst o tom, jak se této„dani“ vyhnout. O instalaci Linuxu na nových modelech notebooků se dočtete na adrese*http://tuxmobil.org/recent_linux_laptops.html.* Informace pro Českou republiku jsme uvedli jižv kapitole „Funkčnost v Linuxu“.

Distribuce pro notebook

Požadavky

V dokumentu Battery-Powered HOWTO (*http://tldp.org/HOWTO/Battery-Powered/*) naleznete následující doporučení[1]:„Vzkaz autorům distribucí: Jste-li tvůrci distribuce, děkujeme za přečtení následujících doporuče-ní. Obliba notebooků se stále zvyšuje, většina linuxových distribucí však pro použití na přenos-ných počítačích stále není dobře vybavena. Proveďte ve své distribuci potřebné úpravy a přičiňtese tak o to, aby se tato výzva stala bezpředmětnou.

Instalační procedura by měla obsahovat konfiguraci optimalizovanou pro notebooky. Ani *mini-mální instalace* není často dost malá. Je spousta věcí, bez kterých se uživatel notebooku nacestách obejde. Nepotřebuje tři různé verze editoru vi. Na mnoha přenosných systémech nenívůbec zapotřebí podpora tisku.

Nezapomeňte v dokumentaci popsat *specifické problémy při instalaci na notebook* – například jak distribuci nainstalovat, není-li k dispozici CD/DVD mechanika.Doplňte lepší *správu napájení* a transparentní *podporu PCMCIA*. Přidejte jádro s *podporou apm*a alternativní sadu ovladačů PCMCIA, které si uživatel může podle potřeby nainstalovat. Doplňtedistribuci o *balíček apmd*. Nezapomeňte také na podporu IrDA a USB.
Doplňte podporu pro dynamické *přepínání konfigurace sítě*. Řada notebooků cestuje mezi místy s různými nastaveními sítě (např. domácí síť, síť v práci, na univerzitě a podobně).Přidejte pohodlný *nástroj pro navazování PPP spojení* s telefonním seznamem, který nebudespouštět více instancí PPP démona, pokud na něj klepnete dvakrát (jako to dělá například user-net v RedHatu). Užitečné bude, pokud takový nástroj zobrazí rychlost připojení a další statistiky.Vhodným nástrojem s automatickou detekcí modemů a PPP služeb je například wvdial, vyvinutýna *http://open.nit.ca/wiki/*.“

Na serveru TuxMobil naleznete řadu odkazů na zprávy o instalaci Linuxu na různé notebooky –viz *http://tuxmobil.org/mylaptops.html.* Jsou řazeny podle výrobce notebooku a linuxové distribu-ce. Jsou k dispozici specifické kategorie pro následující distribuce:

Debian, *http://tuxmobil.org/debian_linux.html*
Gentoo, *http://tuxmobil.org/gentoo_mobile.html*
RedHat, *http://tuxmobil.org/distribution_linux_laptop_redhat.html*

[1] Poznámka českého vydavatele: Uvedený seznam není zcela aktuální, protože podpora notebooků se výrazně zlepšila a stále se zlepšuje s novými verzemi distribucí. Mnoho problémů nového hardwaru odstraníte tak, že použijete co nejaktuálnější distribuci.

- openSUSE (SuSE), *http://tuxmobil.org/distribution_linux_laptop_suse.html.*

Ubuntu, *http://tuxmobil.org/distribution_linux_laptop_ubuntu.html.*
SlackWare, *http://tuxmobil.org/distribution_linux_laptop_slackware.html.*
Mandriva (Mandrake), *http://tuxmobil.org/distribution_linux_laptop_mandrake.html.*
Minix, *http://tuxmobil.org/mobile_minix.html.*
Různé verze BSD, *http://tuxmobil.org/mobile_bsd.html.*

Některé texty jsou k dispozici i v jiných jazycích, *http://tuxmobil.org/lang.html.*

Doporučení

Distribuce Debian GNU/Linux, *http://www.debian.org/*, má asi nejpropracovanější možnosti pro instalaci na notebooky. Instalační nástroj této distribuce je velmi pružný a instalační proces je dobře zdokumentovaný, zejména co se týče metod užitečných právě pro notebooky. Binární sou-bory jsou „stripnuté“, a tedy malé. Existuje konference *debian-laptop* s prohledávatelným archi-vem. A samozřejmě je tato distribuce zdarma.

Pochopitelně existují i jiné distribuce, které s notebooky také vycházejí dobře.

Instalace

Související dokumentace

CDROM HOWTO, *http://tldp.org/HOWTO/CDROM-HOWTO/.*
CD Writing HOWTO, *http://tldp.org/HOWTO/CD-Writing-HOWTO.html.*
Config HOWTO, *http://tldp.org/HOWTO/html_single/Config-HOWTO/.*
Diskless HOWTO, *http://tldp.org/HOWTO/Diskless-HOWTO.html.*
Installation HOWTO, *http://tldp.org/HOWTO/Installation-HOWTO/.*
Pre-Installation Checklist HOWTO, *http://tldp.org/HOWTO/Pre-Installation-Checklist/-index.html.*
Update HOWTO, *http://tldp.org/HOWTO/Update.html.*
Hard Disk Upgrade HOWTO, *http://tldp.org/HOWTO/Hard-Disk-Upgrade/.*
Linux Installation and Getting Started, *http://www.tldp.org/LDP/gs/gs.html.*
Installing Debian GNU/Linux For Intel x86, http://www.debian.org/releases/stable/ i386/install.
Install From Zip HOWTO, *http://tldp.org/HOWTO/Install-From-ZIP.html.*
ZIP Drive HOWTO, *http://tldp.org/HOWTO/ZIP-Drive.html.*

Prerekvizity – BIOS, parametry bootování, rozdělení disku

BIOS

Začínáte-li s novou instalací, měli byste vyjít ze standardního nastavení BIOSu. Teprve pokud něco nebude fungovat, můžete zkusit nastavení měnit. Známým potížistou je například nastavení Plug-and-Play, často schované pod různými názvy. Další informace naleznete také dále v kapitole věnované hardwaru.

Parametry bootování

Existuje mnoho voleb ovlivňujících bootovací proces, které mohou mít na notebooku různé efek-ty – například apm=on|off nebo acpi=on|off. Podrobnosti naleznete v dokumentu Boot Prompt HOWTO, *http:///tldp.org/HOWTO/BootPrompt-HOWTO.html,* a v dokumentaci jádra v souboru /usr/src/linux/Documentation/kernel-parameters.txt.

Rozdělení disku

Rozdělení disku na oddíly lze provést mnoha sofistikovanými způsoby. Zde uvedeme jenom několik základních myšlenek. Lze předpokládat, že na notebooku mohou být důvody pro duální instalaci Linuxu a Windows 9x/NT (například aktualizace firmware PCMCIA zařízení nebo BIOS). V závislosti na konkrétních potřebách a vlastnostech konkrétního notebooku mohou být zapotřebí různé diskové oddíly:

BIOS, některé BIOSy například na noteboocích Compaq používají vlastní diskový oddíl,
uspání na disk, funkce podporovaná některými BIOSy,
- odkládací oddíl pro Linux,
- odkládací oddíl pro Windows,

- systémový oddíl pro Linux,
- oddíl /home pro osobní data (kvůli bezpečnosti zvažte použití šifrovaného souborového systému, podrobnosti naleznete v dalším textu),

společný oddíl pro sdílení dat mezi Linuxem a Windows,
malý (cca 32MB) bootovací oddíl pro yaBoot (zavaděč Linux/PPC) ve formátu HFS MacOS.

Uvedený přehled není zdaleka vyčerpávající. Doporučujeme přečíst si příslušné dokumenty HOWTO, například *http://tldp.org/HOWTO/Partition/.*

Linuxové nástroje pro přerozdělení disku GNU parted

Program GNU parted, *http://www.gnu.org/software/parted/index.shtml,* umožňuje vytvářet, rušit, měnit velikost a kopírovat diskové oddíly. Podporuje více souborových systémů: ext2/ext3, fat (fat16 a fat32), linuxové odkládací oddíly, MS-DOS, Macintosh, PC98 a další (aktuální informace najdete na webových stránkách). Pro souborové systémy NTFS můžete použít též ntfsresize,
http://mlf.linux.rulez.org/mlf/ezaz/ntfsresize.html.

ext2resize

Program ext2resize, *http://ext2resize.sourceforge.net/,* dokáže měnit velikost souborových systé-mů ext2 a ext3, vychází z uvedeného projektu parted. Dokáže ověřit, zda je nová požadovaná veli-kost oddílu vyhovující (tedy zda se na ni vejdou stávající uložená data).

fixdisktable

V konferenci *linux-kernel@vger.rudgers.edu* se před časem objevila zpráva o programu na rekon-strukci diskových oddílů. Nikdy jsme jej nepoužívali a nezkoumali. Mohl by být užitečný v přípa-dě, že vám programy jako Ranish Partition Manager nebo Partition Magic poškodí údaje o disko-vých oddílech. Informace o tomto programu naleznete na adrese *http://bmrc.berkeley.edu/people/chaffee/fat32.html.*

Úskalí

Než začnete měnit rozdělení oddílů na pevném disku, zjistěte si stávající rozložení disku. Zejmé-na se podívejte po skrytém diskovém prostoru nebo speciálních oddílech pro uspání na disk nebo hibernaci. Některé notebooky mají speciální oddíl s BIOSem (například Compaq Armada 1592DT). Podívejte se do dokumentace a hledejte věci jako PHDISK.EXE, uspání na disk, diagnostické nástroje a podobně.
Patrick D. Ashmore, *http://www.procyon.com/~pda/lphdisk/,* vytvořil nástroj, který připraví oddíl pro hibernaci na disk na noteboocích s BIOSem Phoenix NoteBIOS. Tento nástroj nesouvisí s funkcí uspání na disk APM, pokud tedy oddíl pro upsání již máte, měli byste být schopni jej použít s jakýmkoliv operačním systémem, který podporuje uspání prostřednictvím APM.

Platí, že jakmile jednou vyměníte disk, rozšíříte paměť nebo změníte rozdělení disku, pak se buď budete muset bez uspání na disk obejít anebo nabootovat DOS a použít program PHDISK.EXE dodávaný s notebookem nebo stažený přímo od Phoenix Technologies. Uživatelé Linuxu jsou těchto problémů ušetřeni, protože pomocí programu lphdisk mohou potřebný oddíl vytvořit přímo v Linuxu.

Přečtěte si také kapitolu o dosových nástrojích pro rozdělení disku.

Multiboot

V kapitole „Na cestách“ naleznete informace o způsobech bootování více operačních systémů z jednoho pevného disku.

Metody instalace notebooku

Citujeme z dokumentu Battery Powered HOWTO, *http://tldp.org/HOWTO/Battery-Powered/:* „Insta-lace a použití Linuxu na notebooku je většinou bezproblémové, takže to klidně vyzkoušejte. Na rozdíl od jiných operačních systémů Linux i nadále podporuje velmi starý hardware, takže s jeho pomocí můžete reinkarnovat i některé velmi staré modely.“

Jednou z velkých výhod Linuxu je množství pružných instalačních voleb, které zde nebudeme podrobně popisovat. Místo toho se zaměříme na specificky notebookové metody, které jsou zapotřebí za některých speciálních okolností.

Většina současných distribucí podporuje instalační metody vhodné i pro notebooky, včetně insta-lace z externí CD-ROM nebo přes PCMCIA kartu a NFS či SMB. Přečtěte si dokumentaci ke kon-krétní distribuci a podívejte se také na výše vyjmenované návody a HOWTO.

Bootovací disketa a CD/DVD-ROM – tradiční způsob

Na moderních noteboocích by měla bez potíží fungovat tradiční instalační metoda – bootovací dis-keta, pomocná disketa a zbytek na CD-ROM nebo DVD. Podmínkou je samozřejmě existence dis-ketové mechaniky a CD mechaniky. I tak ale můžete na některých noteboocích narazit na potíže, například pokud *nelze použít disketovou a CD mechaniku současně* nebo pokud je disketová mechanika přístupná *pouze jako PCMCIA zařízení* – například Toshiba Libretto 100. Některé note-booky podporují bootování a následně celou instalaci přímo z CD mechaniky, například SONY VAIO (postup je popsán v dokumentu *http://tldp.org/HOWTO/VAIO+Linux.html*). Nezapomeňte v BIOSu ověřit nastavení bootování z CD mechaniky a ověřte si, že je distribuce dodávána na boo-tovatelném disku.

Na některých noteboocích se vám podaří nabootovat pouze jádra uložená v obrazu *zImage*, obra-zy *bzImage* nefungují. Je to například případ notebooku IBM Thinkpad 600 a Toshiba Tecra. Ně-které distribuce (například Debian GNU/Linux) obsahují speciální bootovací diskety pro tyto modely notebooků nebo obecně pro modely s malou pamětí.

Instalace z CD/DVD – obvyklý způsob

Novější notebooky dokážou nabootovat linuxovou distribuci z bootovatelných CD či DVD disků. Tak je možné systém nainstalovat i bez disketové mechaniky. Pokud je CD/DVD mechanika dostupná pouze jako PCMCIA zařízení (například v notebooku SONY VAIO PCG-Z600TEK), pře-čtěte si kapitolu o instalaci z PCMCIA zařízení.

Instalace z dosového či Windows oddílu na stejném stroji

Následující stručný popis obsahuje návod pro instalaci Debianu z CD disku pod Dosem bez nut-nosti použít bootovací či jinou pomocnou disketu. Tento postup je vhodný zejména pro notebo-oky, které neumožňují současné připojení disketové a CD mechaniky, nebo pro notebooky, kde se tyto mechaniky připojují jako PCMCIA zařízení. Postup je převzat z návodu pro instalaci distri-buce Debian, *http://www.us.debian.org/releases/stable/installmanual.*

Ze zrcadla Debianu si stáhněte soubory resc1440.bin, drv1440.bin, base2_1.tgz, root.bin, linux, install.bat a loadlin.exe a uložte je do adresáře na dosovém oddí-lu.
Nabootujte do Dosu (nikoliv do Windows) bez zavádění jakýchkoliv ovladačů. Dosáhnete toho tak, že ve správné fázi bootovacího procesu zmáčknete F8.
Z adresáře s uloženými soubory spusťte install.bat.
Restartujte systém a nainstalujte zbytek distribuce, v této fázi budete schopni použít tech

nologie jako PCMCIA, PPP a další. S nezbytnými úpravami by mohl tento postup fungovat i pro jiné distribuce.

Instalace z druhého počítače pomocí linuxové mikrodistribuce na disketě Úvod

Vzhledem k minimálním (a v některých případech nulovým) nárokům na diskový prostor jsou linuxové mikrodistribuce velmi vhodné právě na notebooky, zejména pokud jste nuceni používat služební notebook s nainstalovanými Windows. Lze je také použít k instalaci prostřednictvím dru-hého nelinuxového počítače. Existuje několik linuxových mikrodistribucí, které bootují z jedné či dvou disket a běží v ramdisku. Jejich seznam naleznete v příloze A.

Následujícím postupem jsme s distribucí muLinux (*http://mulinux.dotsrc.org/*) klonovali systém z notebooku HP OmniBook 800 na Compaq Armada 1592DT. V současné době muLinux nepod-poruje PCMCIA, v případě potřeby proto raději použijte TomsRtBt. Ten ale pro změnu nepodpo-ruje *PPP*, podporuje však *slip*. (Od verze 7.0 už muLinux obsahuje doplněk s podporou PCMCIA).

Popíšeme si, jakým způsobem zkopírovat již existující diskový oddíl, podobným postupem však můžete realizovat i vlastní instalaci. Typicky se budeme snažit rozběhnout instalaci přes NFS, která je podporována většinou distribucí. Pokud zdrojová data

nemáte na linuxovém stroji, můžete místo NFS použít protokol SMB, který je muLinuxem rovněž podporován.

Prerekvizity

Potřebujete dva počítače vybavené Linuxem. Na notebooku (cílovém počítači), na němž chcete nainstalovat Linux, nabootujte muLinux z diskety. Na druhém počítači (serveru, zdroji) může běžet jakýkoliv Linux, případně na něm rovněž můžete nabootovat muLinux. I přes nízkou přenosovou rychlost jsme pro přenos použili sériový null-modemový kabel, protože je levný. Můžete postup upravit a data přenášet přes síťovou kartu PCMCIA a křížený kabel nebo paralelní „null-modemo-vý“ kabel a PLIP. Jako základní protokol jsme použili PPP, můžete ale použít i SLIP. Pro samotný přenos dat posloužil nc. (Na některých distribucích jej najdete pod názvem netcat. Můžete ale použít i ftp, tftp, rsh, ssh, dd, rcp, kermit, NFS, SMB nebo jiné metody přenosu.)
Mezi základní požadavky patří:

Dobrá znalost práce s Linuxem. Pokud nebudete přesně vědět, co děláte, mohli byste si poškodit stávající instalaci.
Null-modemový kabel.

Zdrojový počítač

Na zdrojovém počítači proveďte následující postup (upozornění: IP adresa, číslo portu, diskový oddíl a terminál jsou pouze příklady):

1. Upravte soubor /etc/ppp/options tak, aby obsahoval pouze následující řádky:

```
/dev/ttyS0
115200
passive
```

Používáte-li muLinux verze 3.x, můžete použít pohodlnější příkaz setup -f ppp.
Příkazem pppd spusťte PPP démon.
Nastavte síťové zařízení PPP: ifconfig ppp0 192.168.0.1.
Přidejte výchozí bránu: route add default gw 192.168.0.1.
Spusťte příkaz ping 192.168.0.2, i přestože cílová strana není dosud nastavena.
Po nastavení cílové strany zahajte přenos z další konzoly příkazem cat /dev/hda2 | gzip -c | nc -l -p 5555. (Na další konzolu se přepnete klávesami Alt+F*x*.)
Po skončení přenosu ukončete ping (killall ping).

Cílový počítač

Na cílovém počítači zadejte následující příkazy:

1. Upravte soubor /etc/ppp/options tak, aby obsahoval pouze následující řádky:

```
/dev/ttyS0
115200
passive
```

Používáte-li muLinux verze 3.x, můžete použít pohodlnější příkaz setup -f ppp.
Příkazem pppd spusťte PPP démon.
Nastavte síťové zařízení PPP: ifconfig ppp0 192.168.0.2.
Přidejte výchozí bránu: route add default gw 192.168.0.2.
Ověřte spojení na zdrojový počítač příkazem ping 192.168.0.1.
Přepněte se na jinou konzolu a aktivujte příjem dat ze zdrojového počítače příkazem nc

```
192.168.0.1 555 | gzip -dc >/dev/hda4.
```

Přenos 400 MB bude trvat přibližně 6 hodin.
Po skončení přenosu jej ukončete stiskem Ctrl+C. Lze se tomu vyhnout (netestováno) přidáním parametru -w 3 v příkazu nc na cílovém počítači (nc -w 3 192.168.0.1 5555 | gzip -dc >/dev/hda4).
Po skončení přenosu ukončete ping (killall ping).

Nastavení cílového počítače po skončení přenosu

Upravte soubor /etc/fstab.
Upravte soubory /etc/lilo.conf a /etc/lilo.msg a spusťte lilo.
Nastavte jádru nové kořenové zařízení: rdev image kořenové_zařízení.

Různé

Místo příkazu gzip můžete použít příkaz bzip2 (netestováno).
Na muLinuxu nejsou k dispozici démony rshd, sshd ani ftpd, proto jsme museli vytvořit vlastní přenosový mechanismus pomocí nc.
PPP démony na obou strojích je nutné spustit poměrně rychle po sobě, jinak se spojení rozpadne.
Bylo by vhodné optimalizovat přenosovou rychlost, možná pomohou volby asyncmap 0 nebo local.
Postup máme vyzkoušený pouze s cílovým oddílem větším než zdrojovým, můžete také vyzkoušet místo příkazu cat použít příkaz dd. Další možností (netestovanou) je na cílovém počítači spustit v kořenovém adresáři nc -l -p 5555 | bzip2 -dc | tar xvf -a na zdrojo-vém počítači spustit v kořenovém adresáři tar cvf - . | bzip2 | nc -w 3 192.168.0.2 5555. Tímto způsobem by se měla zkrátit i přenosová doba, protože se přenášejí jen skutečně používané oblasti disku.
Cílový oddíl nesmíte mít připojen příkazem mount.

Instalace z druhého počítače pomocí 2,5” diskového adaptéru

Adam Sulmicki, *adam@cfar.unc.edu*, doporučil následující trik: „Většina pevných disků v notebo-ocích se dá vymontovat, i když to ne vždy musí být jednoduché. Můžete si koupit levný 2,5" IDE adaptér a disk z notebooku dočasně připojit do klasického PC a Linux nainstalovat na něm. Disk můžete připojit jako primární i jako sekundární, pak si ale musíte dát pozor, aby lilo zapisovalo na správný oddíl. Nezapomeňte také na nastavení stejného způsobu mapování, jaký používá note-book (tj. LBA, LARGE nebo CHS).“ Další informace můžete získat v dokumentu *http://tldp.org/HOWTO/Hard-Disk-Upgrade/index.html.* Tímto způsobem můžete překopírovat exi-stující oddíl anebo spustit úplnou instalaci.

Instalace z PCMCIA zařízení

Nemáme k dispozici notebook s PCMCIA disketovou mechanikou (jako je například Toshiba Lib-retto 100), takže následující popis nemáme ověřený. Podívejte se do kapitoly o bootování z PCMCIA zařízení v dokumentu *http://pcmcia-cs.sourceforge.net/ftp/doc/PCMCIA-HOWTO.html.* Nemáme také ověřeno, zda je možné bootovat z PCMCIA pevného disku.

Pokud se vám nicméně podaří nabootovat z diskety a notebook má k dispozici PCMCIA slot, může-te pomocí vhodné PCMCIA karty připojit jiný počítač, externí SCSI zařízení, externí CD či ZIP mecha-niku a podobně. Tyto metody bývají typicky popsány v dokumentaci k instalované distribuci.

Notebook Sony Vaio (PCG-Z600) má externí USB disketovou mechaniku a externí PCMCIA CD mechaniku. Z CD mechaniky se vám podaří nabootovat, pak už ji ale Linux nerozezná a nebu-dete z ní moci pokračovat v instalaci. Tento problém můžete vyřešit parametrem ide2=0x180,0x360 nebo 0x180,0x386 zadaným v promptu LILO.

Instalace ze ZIP mechaniky na paralelním portu

Tuto metodu nemáme ověřenu. Popisuje ji dokument *http://tldp.org/HOWTO/Install-From-ZIP.html.* Nevíme také, nakolik je tato metoda podporována různými distribucemi nebo mikrodistribucemi. Lze předpokládat, že s jejím užitím bude trochu práce.

Jeremy Impson, *jdimpson@acm.org*, píše: „Instaloval jsem Red Hat 6.1 na notebook Libretto 50CT, který má pouze PCMCIA disketovou mechaniku. (Mimochodem, stávající ovladač PCMCIA floppy ji nepodporuje příliš dobře, bylo nutné jej patchovat.)

Linux z této mechaniky nabootoval, po zavedení jádra už s ní však nedokázal pracovat. Libretto má pouze jeden PCMCIA slot (novější modely už mají dva) a neměl jsem k dispozici ani repliká-tor portu. Nebylo proto možné po nabootování z diskety pokračovat v instalaci ze vzdáleného počítače.

Stáhl jsem si proto ZipSlack (Slackware navržený ke spuštění ze ZIP disku) a na druhém počítači jsem jej na ZIP disk nahrál. Připojil jsem mechaniku k notebooku (přes paralelní port) a naboo-toval Slackware z diskety v PCMCIA mechanice. Po nabootování jsem odpojil disketovou mecha-niku a připojil a nastavil síťovou PCMCIA kartu. Jádro je v tomto okamžiku v paměti a používá souborový systém na ZIP disku.

Rozdělil jsem a naformátoval pevný disk a přes FTP stáhl instalační obrazy Red Hatu (na diskový oddíl, ze kterého bude nakonec /home). To je samozřejmě klíč k úspěchu – potřebujete mít dosta-tek diskového prostoru, abyste mohli mít na disku jak instalační obrazy, tak prostor, na nějž bude-te systém instalovat.

Poté jsem ukončil ZipSlack, nabootoval z instalačního disku Red Hatu z disketové mechaniky a pokračoval v instalaci ze souborů uložených už na pevném disku.“

Instalace z CD mechaniky připojené přes paralelní port (MicroSolutions BackPack)

Instalovali jsme Linux pomocí paralelní CD mechaniky MicroSolutions BackPack. Linux ji plně podporuje a s instalací nebyl žádný zásadní problém. Do jádra 2.0.36 byla podporována vlastním modulem bpck, v dalších verzích je ovladač součástí obecnějšího IDE adaptéru pro sériový port (modul paride, který samozřejmě potřebuje další moduly nižší úrovně, v tomto případě stále modul bpck).

V instalaci Red Hatu 5.x je modul bpck přímo součástí instalační konfigurace, takže při instalaci stačí z nabídky *Other CD-ROMs*

vybrat zařízení BackPack a pak nastavit potřebné parametry (i když autoprobe většinou funguje spolehlivě).

V Red Hatu 6.x (který používá jádro 2.2 a paride) byla podpora BackPack odstraněna. Pokud tedy potřebujete tuto distribuci instalovat z podobného zařízení, musíte upravit bootovací disk (doplnit na něj nezbytné moduly) a pak by instalace měla fungovat bez problémů.

Federico Pellegrin vytvořil upravený bootovací disk Red Hatu, který obsahuje ovladače všech paralelních CD-ROM zařízení podporovaných distribučním jádrem, a neměl by tedy být problém z jakéhokoliv takového zařízení systém nainstalovat.

Počínaje distribucí Red Hat 6.2 obsahuje instalace ještě doplňkový disk, na němž ovladače para-lelních zařízení jsou. Musíte si tedy standardním způsobem vytvořit disk s ovladači (obraz disku paride.img naleznete v adresáři images/drivers) a pak jej vložit do mechaniky ve chvíli, kdy
o něj instalátor požádá.
Lze předpokládat, že pomocí podobných zařízení by neměl být problém nainstalovat i jakoukoliv jinou distribuci, potřebujete prostě přidat a nastavit příslušné moduly hned na začátku instalačního procesu.

Měli byste si dát pozor na nastavení režimu paralelního portu (ECP, EPP, Output only, PS/2), protože při některých nastaveních může notebook nečekaně zamrzat, nebo dokonce poškodit data. V jiných režimech může být komunikace se zařízením pro změnu velmi pomalá. Na mém notebooku se ukázalo jako nejvhodnější nastavení PS/2, ale nemusí to platit vždy.

Autorem této kapitoly je Federico Pellegrin. Další informace naleznete také v dokumentu *http://tldp.org/HOWTO/CDROM-HOWTO/.*

Instalace z druhého počítače přes paralelní port (PLIP)

Následující návod zaslal Nathan Myers, *ncm@cantrip.org*: „Řada distribucí podporuje instalaci přes síť prostřednictvím FTP, HTTP nebo NFS. Je poměrně běžné, že dnešní notebooky mají jenom jedi-ný PCMCIA slot, do kterého už musí být připojena bootovací disketová mechanika. Obraz booto-vací diskety typicky neobsahuje ovladače ani pro samotnou mechaniku, ani pro subsystém PCMCIA. Jediným dostupným síťovým rozhraním tak může být paralelní port.

Instalace přes paralelní port s využitím protokolu PLIP byla vyzkoušena na distribuci Red Hat. Po-třebujete paralelní kabel *Laplink*, který je k dostání v každém počítačovém obchodě. Podrobnos-ti o nastavení spojení naleznete v dokumentu *http://tldp.org/HOWTO/PLIP.html.* Konkrétně insta-lace Red Hatu vyžaduje, aby byla vzdálená strana PLIP spojení nastavena na použití protokolu ARP (zjevně proto, že Red Hat v instalátoru používá ovladač DOS). Na zdrojovém počítači pak insta-lační CD vyexportujte do NFS nebo je připojte někam, kde bude přístupné prostřednictvím démo-nů FTP či HTTP.“

Dokument *http://tldp.org/HOWTO/PLIP.html* Gillese Lamirala popisuje, jak Linux nainstalovat na počítači, který nemá ani síťovou kartu, ani CD mechaniku, pouze disketovou mechaniku a vzdá-lený server NFS připojený paralelním kabelem.

Instalace z USB zařízení (klíčenka, CD, DVD, disketa)

Pokud BIOS podporuje bootování z USB zařízení, je možné Linux nainstalovat tímto způsobem.
Až na některé staré modely podporuje tuto funkci většina notebooků vybavených USB porty. Nejprve musíte BIOS nastavit tak, aby bootoval z USB zařízení. Někdy je možné stiskem určité klá-vesy či kombinace kláves v době bootování zvolit, z jakého zařízení se má bootovat.
Dále musíte přenést Linux na bootovací médium (řekněme na USB klíčenku) a nastavit ji jako boo
tovatelnou. Přesně pro tyto účely existuje několik speciálních distribucí Linuxu, například: Feather Linux, *http://featherlinux.berlios.de/about.htm,* je distribuce běžící kompletně z CD nebo USB, zabírající méně než 64 MB místa. Jde o předělaný Knoppix (založený na Debian GNU/Linu-xu), obsahující většinu programů, které normální člověk na pracovním počítači běžně používá. Důležitý je dokument *http://featherlinux.berlios.de/usb-instructions.htm,* popisující instalaci Feat-her Linuxu na USB zařízení.

Partboot, *http://www003.upp.so-net.ne.jp/tshiono/partboot-usb/,* je určen pro USB disketové mechaniky a upraven pro instalace na noteboocích (obsahuje nástroje pro změnu rozdělení disku, podporu PCMCIA a další).

Damn Small Linux (DSL), *http://www.damnsmalllinux.org/,* je live distribuce pro 50MB CD veli-kosti kreditní karty. Navzdory minimální velikosti obsahuje funkční a snadno použitelné pracovní prostředí.
Použít lze také distribuce Slax (*http://www.abclinuxu.cz/clanky/navody/slax-na-usb*), český Danix (*http://www.danix.cz/vyvojari/spousteni-z-usb-flashdisku*). Existuje také speciální verze Mandriva Linuxu dodávaná přímo s USB diskem – Mandriva Flash (*http://www.mandriva.com/en/ individuals/products/node_3482).*

Instalace přes síové rozhraní

Většina moderních notebooků má integrovanou síťovou kartu a velmi snadno tak lze instalovat pomocí protokolu PXE. Tento postup je užitečný zejména v případech, kdy není k dispozici CD či DVD mechanika.

Příprava zdrojového počítače

My jsme při instalaci používali na zdrojovém počítači CD Knoppixu. Stačí pouze zapnout termi-nálový server (KNOPPIX -> Server-Dienste -> Terminal-Server KNOPPIX-Services-Start -> KNOPP-IX Terminal Server). U většiny notebooků budou fungovat výchozí síťové ovladače. Vypněte bez-pečnostní volby, jinak se nebudete moci na cílový počítač přihlásit jako *root*. Samozřejmě existuje i celá řada jiných způsobů, jak na zdrojovém počítači zprovoznit protokol PXE, není nutné pou-žívat Knoppix. Měl by fungovat i protokol EtherBoot, nemáme to však vyzkoušeno.

Příprava cílového počítače

Hledejte v BIOSu nastavení jako „NetBoot Option“ a zapněte je. Zapněte počítač a zvolte booto-vání ze síťového zařízení. Dosáhnete toho buď stisknutím určité klávesy ve vhodné fázi bootova-cího procesu nebo tak, že síťové rozhraní v BIOSu nastavíte jako první bootovací. Měl by naběh-nout Knoppix. Otevřete shell a příkazem su se přepněte na účet *root*. Instalaci na pevný disk zahá-jíte příkazem knx-hdinstall (Knoppix<=3.3) nebo knoppix-installer (Knoppix>3.3).

Instalace přes VNC

Ptáte se, proč instalovat notebook přes protokol VNC? Jeden hypotetický důvod by byl. Můžete mít notebook s poškozenou klávesnicí a instalaci tak budete provádět ze vzdáleného počítače. Klávesnice ovšem nesmí být zničená úplně, protože potřebujete být schopni alespoň spustit VNC. Novější verze distribuce openSUSE jsou pro tento typ instalace již vybaveny, podívejte se do doku-mentace. A distribuce Mandriva Linux podporuje při síťové instalaci přesměrování instalačního dis-pleje na vzdálený X server pomocí parametru display. Nejedná se sice o VNC přístup, ale mož-nosti jsou podobné, viz českou dokumentaci k Mandriva Linuxu.

Instalace Linuxu na slabších strojích

Pokud máte počítač s méně než 8 MB paměti a pokoušíte se o instalaci přes NFS, asi vás zastaví chybové hlášení „fork: out of memory“. Problém můžete vyřešit tak, že pomocí programu fdisk vytvoříte odkládací oddíl (program fdisk by měl být na instalační disketě, případně můžete pou-žít některou z výše popsaných minidistribucí). Pak znovu nabootujte z instalační diskety. Před nastavením NFS spojení se přepněte na jinou konzolu (například stiskem Alt+F2) a zadejte příkaz swapon /dev/xxx, kde *xxx* je název odkládacího oddílu. Tento tip zaslal Thomas Schmaltz.

Bruce Richardson vytvořil dokument *http://tldp.org/HOWTO/4mb-Laptops.html* o instalaci moderní distri-buce (konkrétně Slackware 7.0) na notebooku se 4 MB paměti a méně než 200 MB diskového prosto-ru. Dalším podobným dokumentem je *http://www.xs4all.nl/~Elennartb/rescuedisk/index.html* od L. C. Benschopa.

Instalace Linuxu na Apple Macintosh PowerBook a iBook

Macintosh PowerBook je v dnešní době vybaven CD/DVD mechanikou, nemá však disketovou mechaniku. Linuxové distribuce pro PPC nicméně podporují bootování a instalaci z CD bez nutnosti použít disketu.

Občas se po nabootování instalačního disku na PowerBooku stane, že obrazovka zčerná. Jednoduše to opravíte tak, že na klávesnici zvýšíte jas (z nějakých důvodů občas dochází ke zresetování nastavení jasu na nulu).

Pokud máte některý z nejnovějších modelů PowerBooku, nemusí být podporován jádrem na insta-lačním CD. Tento problém můžete vyřešit tak, že nabootujete novější jádro, které si stáhnete na pevný disk a použijete ramdisk, instalace jednotlivých balíčků bude i nadále probíhat z CD. (Podí-vejte se na online dokumentaci k programům yaBoot nebo BootX, což jsou zavaděče pro Linux/PPC. yaBoot je pro nové modely vhodnější.)
Je také možno nabootovat a instalovat z nativního Macintosh oddílu (HFS) na pevném disku. Výsledky instalace Linuxu na různých modelech naleznete na adrese *http://tuxmobil.org/ apple.html.*

Hromadná instalace IDE adaptér z 2,5” na 3,5”

Pokud máte k dispozici IDE redukci z 2,5" na 3,5", můžete nainstalovat jeden notebook a na stol-ním počítači naklonovat jeho pevný disk na dalších 99 notebooků. Můžete použít dosový nástroj GHOST (dobře zvládá souborový systém ext2), anebo pokud je stolní počítač linuxový, můžete použít přímo tar. Pak už potřebujete na každém notebooku jen nabootovat z diskety, přeinstalo-vat lilo a změnit název a IP adresu. Redukce je velmi levná, obtížně se ale shání.

SystemImager

Program VA SystemImager, *http://systemimager.sourceforge.net/,* je software, který usnadňuje hro-madnou instalaci Linuxu na podobný hardware. Usnadňuje také distribuci a konfiguraci programů a aktualizaci operačního systému. Jeho prostřednictvím můžete dokonce aktualizovat z jedné verze distribuce na novou. Program lze také použít ke správě obsahu na webových serverech. Nejužitečnější je v situacích, kdy provozujete velký počet identických počítačů. Mezi typické oblas-ti nasazení patří: farmy internetových serverů, výkonné clustery, počítačové učebny nebo firemní počítače, které mají stejnou základní hardwarovou konfiguraci.

Debian GNU/Linux

Podívejte se na dokument *http://www.informatik.uni-koeln.de/fai/*.

SuSE/openSUSE

Balík ALICE, Automatic Linux Installation and Configuration Environment, nabízí správu konfiguračních souborů a šablon založenou na CVS.

Mandriva Linux

Mandriva Linux disponuje nástrojem drakautoinst, pomocí kterého lze jednoduše vytvářet dis-kety na klonování (aktuální) instalace. Pomocí speciálního souboru pro instalační program pak lze během instalace nastavit téměř cokoliv. Další informace najdete na *http://wiki.mandriva.com/ Development/Howto/MDK_Installer.*

Replicator

Replicator, *http://sourceforge.net/projects/replicator/,* je sada skriptů pro automatickou duplikaci instalace Debian GNU/Linuxu z jednoho počítače na druhý. Replicator bere v úvahu rozdíly v hardwaru jednotlivých počítačů (jako je například velikost disku nebo grafická karta) i rozdíly v softwarové konfiguraci (například rozdělení disků na oddíly). Po počátečním nastavení se vytvo-ří bootovací disketa, která vám umožní nainstalovat či přeinstalovat Debian pouhým nabootová-ním z diskety a odpovídáním na otázky.

bpbatch

Dobrou alternativu představuje také nástroj bpbatch, *http://www.bpbatch.org/.*

partimage

Program PartitionImage, *http://www.partimage.org/,* dokáže vytvořit obraz diskového oddílu se souborovým systémem ext2/ext3, ReiserFS a FAT16/32. Obraz je možné komprimovat nástroji GZIP/BZIP2 a rozdělit na více souborů pro rozkopírování na záložní média.

Běžné problémy při instalaci

Problémy se zobrazením (chybějící řádky, široké okraje)

Běžným problémem při instalaci Linuxu na notebook jsou chybějící řádky ve spodní části textové konzoly, takže na obrazovce není vidět poslední příkaz či výzva k přihlášení. Podle konkrétní situ-ace mohou pomoci následující kroky:

Použít framebuffer, tedy použít jádro s podporou framebufferu, a nabootovat s volbou typu `vga=791`. Podrobnosti viz dokument *http://tldp.org/HOWTO/Framebuffer-HOWTO.html.*
Naopak vypnout framebuffer a nabootovat s volbou `vga=normal`. V bootovací výzvě insta-látoru můžete také zkusit zadat `video=vga16:off`.
Jako nouzové řešení často stačí přepnout se na jinou konzolu, například Alt+F2, protože popsaný efekt se většinou týká jen první konzoly.
Ověřte, zda jsou v zavaděči (GRUB či LILO) nastaveny volby bootování VGA a videa. Zkus-te je alespoň částečně vypnout, hledejte volby jako ywrap a podobně.
Zkontrolujte nastavení obrazovky v BIOSu, byla to častá příčina problémů u starších notebooků Toshiba.
Příkazem resize nastavte v systému správnou velikost obrazovky.
Pokud nic z výše uvedeného nepomůže, vytvořte si inicializační skript spouštěný na konci bootovacího procesu, který bude obsahovat příkazy clear a/nebo reset.

Palmtopy, Personal Digital Assistant (PDA), Handheld PC (HPC)

Na začátek uveďme některé zajímavé zdroje informací:

Vřele doporučujeme stránky ARM Linux Russella Kinga, *http://www.arm.linux.org.uk/,* které obsahují informace o PDA s procesory ARM a odkazy na další stránky týkající se Linu-xu na PDA.
Informace o protokolu Virtual Network Computing naleznete na adrese

http://www.realvnc.com/.

Informace o PDA a infračerveném dálkovém ovládání naleznete na stránkách *http://hp. vector.co.jp/authors/VA005810/remocon/remocone.htm.*
Linux můžete provozovat na IBM PC110 (dnes už nevyráběné HPC). Konferenci věnova-nou tomuto tématu naleznete na adrese *http://pc110.ro.nu/mailing.html.*
Na serveru Palmtop.Net, *http://www.palmtop.net/,* naleznete odkazy na témata související s palmtopy.
Stránku na téma Linuxu na PDA a HPC najdete i na serveru Tuxmobil,

http://tuxmobil.org/pda_linux.html.

7. Pro vývojáře aplikací určených pro PDA jsou k dispozici diskusní skupiny codewarrior.embedded, codewarrior.games, codewarrior.linux, codewarrior.mac, codewar-rior.palm, codewarrior.unix a codewarrior.windows.

Itsy

Prototyp modelu Itsy nabídl o poznání větší výpočetní výkon a paměťovou kapacitu než ostatní PDA ve své době, takže na něm bylo možno používat i natolik náročné aplikace jako rozpozná-vání řeči. Byl navržen jako otevřená platforma s cílem usnadnit inovativní výzkumné projekty. Základní model obsahuje flexibilní rozhraní pro připojení vlastních karet, software je založen na Linuxu a standardních GNU nástrojích.

Informace

Výrobcem je Compaq/Digital.

Linuxová PDA

Nejznámějším současným linuxovým PDA je Agenda VR3 (*http://tuxmobil.org/pda_survey_ agenda.html*) společnosti AgendaComputing, iPAQ (*http://tuxmobil.org/pda_survey_compaq.html*) společnosti HP/Compaq, série Zaurus (*http://tuxmobil.org/pda_survey_sharp.html*) od Sharpu a Yopy (*http://tuxmobil.org/pda_survey_samsung.html*) od Samsungu. Kromě iPAQu jsou to všechno opravdové linuxové PDA, vybavené Linuxem už od výrobce.

Pro PDA existuje několik otevřených linuxových distribucí, například QT Embedded (*http://www.trolltech.com/,* instalovaný standardně na Zaurusech), Opie (*http://opie. handhelds.org/cgi-bin/moin.cgi/*) nebo Familiar (*http://familiar.handhelds.org/*). Projekt Gnome Palmtop Environment – GPE (*http://gpe.handhelds.org/*) vytváří grafické rozhraní pro palmtopy s operačním systém GNU/Linux. GPE používá X Window System a GTK+.

Většinu programů pro novější PDA je možno získat jako předkompilované balíčky IPK. K naleze-ní potřebných balíčků můžete použít Zaurus Software Index – ZSI (*http://www.killefiz.de/zaurus/*) nebo ipkgfind (*http://ipkgfind.handhelds.org/*). Balíčky můžete nainstalovat různými způsoby. Jed-ním z nich je přímá instalace přes HTTP spojení, takzvaný *feed*. Tuto službu nabízí i server Tux-mobil na adrese *http://tuxmobil.org/feed.html.*

Kromě známých PDA podporujících Linux se budeme zabývat i porty pro jiná PDA a nástroji k zajištění konektivity s nelinuxovými PDA, mobilními telefony a stolními počítači.

AgendaComputing: Agenda VR3 Informace

Výrobcem prvního čistě linuxového PDA Agenda VR3 byla již neexistující společnost Agenda-Computing.

Samsung: YOPY Informace

1. Výrobcem modelu YOPY je Samsung,
   http://www.sem.samsung.com/eng/ product/digital/pda/index.htm.

2. O YOPY píše i německý Linux Magazin,
   http://www.linux-magazin.de/News/ index_html?newsid=519.

Alternativní stránky, *http://www.theyopy.de/.*

Oficiální stránky, *http://www.yopy.cc/.*

SHARP SL-5000/5500/C700-860/C3x00/6000 též Zaurus

Model Sharp Zaurus SL-5000/5500 sice nebyl prvním linuxovým PDA, jde však o model s největ-ším úspěchem jak v linuxové komunitě, tak mimo ni.

Systém Sharp

Oficiální informace o Linuxu pro Zaurus naleznete na stránkách Sharp Japan (*http://developer.ezaurus.com/* – v japonštině). Nalez-nete zde oficiální jádro, jak kompletní, tak jen potřebné patche. Můžete také získat oficiální souborový systém, ovšem bez prostředí QTopia, *http://qpe.sourceforge.net/.* V dokumentaci se dočtete, jak vyrobit zImage, bootflag a initrd pro nahrání vlastní instalace do ROM Zaurusu. Obraz celé ROM v jediném souboru, takzvaném „ospacku", najdete i na regionálních stránkách, například *http://www.zaurus.de* nebo *http://www.myzaurus.com.* Jádro je dost staré, 2.4.6 s patchi 2.4.6-rmk2 a některými dalšími od Linea (*http://www.lineo.com/*); rmk patche pocházejí z projektu Linux ARM, *http://www.arm.linux.org.uk/.* Kořenový souborový systém má strašlivou strukturu s hromadou symbolických odkazů. Vlastní překlad jádra je možný. Jakmile Zaurus zobrazí „Wait...", zmáčkně-te klávesu /, tím se umožní textové přihlášení namísto spuštění QTopia, která není dostupná, pokud si ji nestáhnete, nepřeložíte a nenainstalujete. Nové uživatele můžete přidávat příkazem addu-ser, který je součástí BusyBoxu, *http://www.busybox.net/.* Ten je součástí oficiálního systému a máte tak dostupnou většinu stan-dardních unixových příkazů.

Komunitní systémy

V současné době jsou nám známy dva funkční systémy – OpenZaurus a Debian.

OpenZaurus

Projekt OpenZaurus, *http://openzaurus.org/wordpress/,* usiluje o vytvoření stejného prostředí, jako nabízí Sharp, ovšem s využitím výhradně svobodného softwaru. V současné době používá stále původní jádro od Sharpu, upravené tak, aby bylo možné FlashROM používat jako RAM a část RAM použít jako ramdisk. Bohužel je ovladač pro čtečku SD karet pouze binární. Sám Sharp se ovšem snaží výrobce (SDCA) přimět k uvolnění zdrojových kódů. OpenZaurus navíc používá souborový systém s klasickou organizací tak, jak ji známe z běžných linuxových systémů. Prostředí QTopia je nahrazeno prostředním Open Palmtop Integrated Environment – OPIE, *http://opie.handhelds.org/cgi-bin/moin.cgi/*, což je fork QTopie vyvíjený nezávisle na Trolltechu. Aplikace určené pro QTopii by měly běžet i v OPIE, neplatí to ale úplně. Například hra Zraycast ve stylu Dooma v QTopii fun-guje, v OPIE však ne. Můžete si stáhnout připravený zImage, bootflag a initrd anebo použít zdro-jové kódy z CVS.

Debian

Aktuální neoficiální verze Debian Zaurus, *http://people.debian.org/~mdz/zaurus/,* je standardní Debian s apt a X Window. Život usnadňuje také jednoduchá verze dpkg, dodávaná jako součást BusyBoxu. Projekt obsahuje několik dalších nástrojů stripnutých tak, aby se vešly na FlashROM. Používá jádro projektu OpenZaurus, tedy stejné jádro jako Sharp. V současné době jsou problé-my s ramdiskem, kalibrací stylusu, uspáváním a probouzením. Jakmile projekt dosáhne větší sta-bility, bude začleněn do EmDebian, *http://emdebian.sourceforge.net,* a budou uvolněny zdrojové kódy. Všechny systémy včetně originálního od Sharpu mají nastaveno americké (nebo německé) rozložení klávesnice. Zdá se, že klávesová mapa je zadrátována přímo v jádře a nejsou k dispozici žádné uživatelské nástroje pro její změnu.

PocketWorkStation

Dále uvádíme některé údaje o projektu PocketWorkStation, *http://www.pocketworkstation.net/*, což je distribuce Debian GNU/Linux určená pro PDA:

Kompletní prostředí Debian GNU/Linuxu se snadným přístupem k mnoha gigabajtům apli-kací. Chcete Konqueror a máte na SD kartě volných 50 MB? Spusťte apt-get install kon-queror, běžte se najíst a po návratu bude prohlížeč připraven k použití.
Obsahuje X Window s možností spouštět většinu linuxových aplikací – podporuje virtuál-ní obrazovky větší, než je fyzická obrazovka, antialiasované změny měřítka a rotace v reál-ném čase, emulace třítlačítkové myši a úplné klávesnice (pokud byste například potřebo-vali aplikaci poslat Ctrl-Alt-Del).
VNC klient fbvnc se stejnými funkcemi jako právě popsané X Window. Ze Zaurusu tak můžete vzdáleně spravovat svá NT.
Běží z jediného adresáře (ideální je 256 MB SD karta), není nutné přeprogramování Flash-ROM nebo modifikace stávajícího systému.
Bez nutnosti rebootování se můžete přepínat mezi QTopií a X Window.

Synchronizace s linuxovým PC

Pracovní prostředí QTopia je zdarma k dispozici u Trolltechu, *http://www.trolltech.com/develo-per/downloads/qtopia/qtopia-gpl.*

Na stránkách naleznete také FAQ popisující postup nastavení Ethernetu přes USB. Dokument není úplně aktuální, protože v novějších verzích ROM Sharp zvý-šil zabezpečení, a tak budete muset rozhraní usb0 nastavit IP adresu 192.168.129.1. Pokud byste chtěli použít ovladač usbdnet, musíte si stáhnout a přeložit vlastní jádro. Poté je možné komuni-kovat mezi QTopií a Zaurusem. Měli jsme potíže s USB vrstvou na pracovní stanici, pomohlo změ-nit ovladač (modul) z uhci na usb-uhci. Mnohem spolehlivější je propojení pomocí ethernetové karty v CF slotu, navíc můžete i nadále používat klávesnici. Nevýhodou samozřejmě je, že nemů-žete mít ve stejné chvíli v CF slotu paměťovou kartu.

Externí sériová klávesnice

Doposud se nám ji nepodařilo rozchodit. Na adrese *http://www.doc.ic.ac.uk/~jpc1/linux/ ipaq/serial.html* naleznete ovladač a jaderný patch pro iPAQ. Vzhledem k tomu, že iPAQ i Zaurus jsou založeny na stejném procesoru, StrongARM, ovladač by měl fungovat i na Zaurusu. Dále potřebujete uživatelský nástroj inputattach, který je na uvedené adrese rovněž k dispozici (ve zdrojové podobě i jako binární soubor pro ARM). Máme k dispozici klávesnici Happy Hacking Lite s konektorem PS/2. Ten pomocí adaptéru převádíme na standardní sériový a pomocí dalšího adaptéru na sériový port Collie. Nejsme si jisti, zda takové uspořádání může být vůbec principiál-ně funkční. Příslušný patch se podařilo aplikovat po drobné úpravě souboru include/linux/serio.h. Po změně konfigurace jádra a překladu jsme nové jádro přenesli na Zaurus. Vznikly moduly newtonkbd.o, serio.o, serport.o a dále stowaway.o z drivers/char/joystick a input.o a keybdev.o z drivers/input. Při spouštění příkazu inputattach musíte zadat příkaz inputattach --newtonkbd /dev/ttyS0, a nikoliv ttySA0, jak říká dokumentace na webu. Z nějakého důvodu nevyhovuje ovladač sériového portu Collie ofici-ální specifikaci jádra StrongARM, která uvádí, že sériové porty jsou přístupné jako /dev/ttySAx. Modul serial_collie.o je standardní součástí jádra od Sharpu, nemusíte tedy zavádět obecný modul serial.o. Při dalších pokusech jsme zkoušeli zavést serial_collie.o jako modul, přes-tože už byl přeložen přímo v jádře. Při pokusu o zavedení nebyla ohlášena žádná chyba, ale systém následně začal nečekaně zamrzat. Zkoušeli jsme inputattach v režimu --dump (je nutné zrušit definice proměnné a nástroj znovu přeložit), vypadalo to, že mezi sériovým portem a klá-vesnicí neprobíhá žádná komunikace. Volání select (man 2 select) skončilo vypršením časového limitu.

Křížový překlad

Jádro

Pro účely překladu jádra, initrd a aplikací potřebujete prostředí pro křížový překlad (cross com-piling), nejvhodnější je samozřejmě GCC. Projekt EmDebian, *http://emdebian.sourceforge.net/,* nabízí balíčky .deb pro GNU/Linux i386. S balíčky g++ a libstdc++-dev mohou být určité problé-my se závislostmi, které lze „vyřešit“ přepínačem --force-depends. Balíček libstdc++-dev má pro-blém s nalezením info souboru, vytvořte symbolický odkaz /usr/share/info/iostream-.ifo.gz na /usr/share/info/iostream-295.info.gz. Návody pro jiné systémy můžete najít na stránkách projektu Linux ARM, *http://www.arm.linux.org.uk/*. Po instalaci vývojového prostře-dí můžete vzít standardní zdrojové kódy jádra, aplikovat příslušné ARM patche a v Makefile změ-nit cílovou architekturu na arm.

Aplikace

Na stránkách projektu QTopia, *http://www.trolltech.com/developer,* naleznete návody pro vývojá-ře aplikací. Obdobné instrukce nabízí i projekt OPIE, *http://opie.handhelds.org/.*

Další nástroje

Werner Schulte nabízí Live CD pro vývoj na platformu OPIE. Na adrese *http://www.uv-ac.de/opiedev/* naleznete HOWTO dokument popisující vytvoření CD a jednotlivé nástroje, které se na něm nacházejí. CD umožňuje křížový překlad programů pro OPIE bez nutnosti instalovat křížový pře-kladač na vlastním počítači.

Dokument *http://www.lucid-cake.net/osx_arm/index_en.html* popisuje nastavení křížového pře-
kladu kompilátorem GCC na platformě Mac OS X.Distribuce DemoLinux,
http://www.pellicosystems.com/demolinux/zdemolinux/index.html, ukazu-je vývojové prostředí pro platformu Sharp Zaurus Personal Mobility Tool či jakoukékoliv jiné ARMzařízení, vybavené systémem Trolltech QPE společnosti Pellico Systems.

Projekt Zaurus Development pro Damn Small Linux, *http://kopsisengineering.com/kopsis/ SharpZaurusSdkDsl,* nabízí prostředí pro křížový překlad binárních souborů pro platformu ARM. Můžete je spustit buď z virtuálního stroje QEMU nebo z Live CD.

Projekt KernelKit, *http://free-electrons.com/community/tools/kernelkit,* je derivát Knoppixu určený pro vývojáře ovladačů zařízení a embedded systémů. Konkrétně obsahuje nástroje pro křížový překlad uClibc pro několik cílových architektur (ARM, i386, MIPS, mipsel, PPC a m68k) a emulá-tory (qemu a SkyEye). Lze jej použít pro demonstraci, výukové účely nebo pro vývojáře, kteří na své stanici nemohou mít nainstalovaný GNU/Linux.

Úskalí

Na spodní straně série Zaurus SL-5x00 je proprietární sériové rozhraní. Můžete si koupit adaptér pro převod na standardní sériové rozhraní, bohužel je adaptér mohutný a nelze následně otevřít klávesnici. Naštěstí do tohoto portu můžete připojit externí klávesnici. Minimálně pak můžete do adaptéru zapojit napájení, takže nemusíte běžet na baterie. Existují i neoriginální adaptéry jiných výrobců, které tímto problémem netrpí.

Model SL-5500 neobsahuje reproduktor, takže pokud chcete přehrávat zvuky, musíte použít konektor pro připojení sluchátek. Bzučák momentálně podporuje pouze 14 různých zvuků defi-novaných v include/asm-arm/sharp_char.h, hledejte text SHARP_BUZ_ALL_SOUNDS.

Zdroje informací

Výrobce: Sharp

http://developer.ezaurus.com/.
http://www.zaurus.com/dev/.

Komunita

ARM Linux, *http://www.arm.linux.org.uk/.*
Embedian, *http://emdebian.sourceforge.net/.*
OpenZaurus, *http://openzaurus.org/wordpress/.*
Sériová klávesnice, *http://www.doc.ic.ac.uk/~jpc1/linux/ipaq/serial.html.*

FAQ, fóra a podobně

1. Sharp Zaurus Hilfe und Support Community (v němčině), *http://www.linuxontour.de/index.php/html/modules/news/.*

Neoficiální Sharp Zaurus SL-5500 FAQ, *http://www.zaurususergroup.org/.*
HandHelds.org – mobilní zařízení, *http://www.handhelds.org/.*

Aplikace, desktopová prostředí

Open Palmtop Integrated Environment (OPIE), *http://opie.handhelds.org/.*
GPE Palmtop Environment, GTK alternativa k OPIE, *http://gpe.linuxtogo.org/.*
QTopia, *http://qpe.sourceforge.net/.*
QTopia-Desktop, *http://www.trolltech.com/developer/downloads.*
Příručka *iPAQ and Zaurus Development using QPE* Wernera Schulte, *http://www.uv-ac.de/ipaqhelp/,* popisuje, jak nainstalovat Familiar Linux a QTopii/OPIE na iPAQu a Zaurusu a jak pomocí těchto nástrojů vyvíjet aplikace.

Seznamy programů

Zaurus Software Index, *http://www.killefiz.de/zaurus/.*
ZaurusSoft, *http://www.zaurusoft.com/.*
IPKGfind, *http://ipkgfind.handhelds.org/.*

Konverze z PalmPilotu na Zaurus

Dokument *http://tuxmobil.org/go2z.html* popisuje konverzní nástroje mezi komerčními operační-mi systémy pro PDA (v současné době jen PalmOS, plánujeme doplnění WinCR/Pocket PC a Epoc) a linuxovými PDA.

Nelinuxová PDA – porty a nástroje

HELIO

HELIO je v současné době dostupné pouze s proprietárním operačním systémem VT. Informace o linuxovém portu naleznete na adrese *http://www.fms-computer.com/.*

Zdroje informací

Výrobcem HELIO je VTech.
2.
Helio PDA.
Existují porty PocketLinuxu, Debianu a RedHatu.
http://www.kernelconcepts.de/helio/.
Křížový překladač VR Org, *http://www.linux-vr.org/.*
Linux-Magazin, *http://www.linux-community.de.*

iPAQ

iPAQ PDA společnosti Compaq/HP je momentálně distribuováno s operačním syste-mem WinCE. Existuje několik distribucí, které je na něm možno použít.

Zdroje informací

1. Výrobcem iPAQu je Compaq/HP,
   http://www.compaq.com.
2. Vše o Linuxu pro iPAQ najdete naadrese *http://www.ipaqlinux.com/.*

Braillův terminál

Stephan Doyon, *http://pages.infinit.net/sdoyon/,* do poštovní konference iPAQu napsal: „S Nicola-sem Pitrem se nám úspěšně podařilo na iPAQ portovat BRLTTY a otestovat funkčnost propojením s BrailleLite 18 přes sériový port. BRLTTY je program, který umožňuje přístup k textové konzole na různých braillových terminálech. BrailleLite je malý elektronický poznámkový blok, který lze použít jako malý braillův displej. Má také klávesy, takže je na něm možné i psát. Máme tedy iPAQ a BrailleLite propojené kabelem a můžeme na iPAQu používat textovou konzolu. Velmi výkonná, a přitom velmi malá konfigurace. Na linuxovém sympoziu v Otavě jsem se pomocí síťové karty v iPAQu přes internetové připojení přihlásil ke svému počítači a pomocí ssh, pine a lynxu jsem si četl poštu! Stejný postup by měl být použitelný i pro jakýkoliv jiný braillův displej nebo jiná PDA.“

Newton Message Pad

Newton Message Pad byl jedním z prvních PDA.

Zdroje informací

Výrobcem zařízení je Apple, *http://www.apple.com/.*
Newton and Linux HOWTO, *http://misf67.cern.ch/~reinhold/Newton/Newton_and_Linux-mini-HOWTO.html.*

PALM-Pilot Zdroje informací

Výrobcem zařízení je 3COM, *http://www.3com.com/.*
Dokument PalmOS-HOWTO Davida H. Silbera,

http://tldp.org/HOWTO/PalmOS-HOWTO/.

PilotLink je nástroj pro přenos dat mezi PalmPilotema linuxovým strojem, XCoPilot je emulátor operačníhosystému PalmPilotu pod Linuxem. Oba nástroje nalez-nete na adrese *http://www.pilot-link.org/.*
ucLinux, *http://www.uclinux.org.*
PalmVNC (*http://www.wind-networks.de/PalmVNC/*) jeimplementace klientské architektury VNC, takže může-te použít linuxový nebo jiný unixový stroj k vytvoření(skromného) prostředí X Window na PalmPilotovi.
Přehled linuxových a BSD aplikací pro PALM nalezne-te na adrese *http://tuxmobil.org/pda_linux_palm.html.*

HandSpring VISOR

HandSpring VISOR je klon PDA PalmPilot.

USB

Z dokumentu /usr/src/linux/Documentation/usb/usb-serial.txt:USB dokovací stanice HandSpring Visor. Webové stránky jsou na adrese *http://usbvisor.sourceforge.net/.*Sériový port HandSpring Visor Platinum je tunelovaný přes USB, takže zaveďte modul usbserial.o s parametry vendor=0x82d product=0x100 (usbmgr.conf). USB se aktivuje po spuštění HotSync synchronizace příkazem pilot-xfer /dev/ttyUSB0 -b -/visor/.

Psion 5

Momentálně máme informace o portu pouze pro Psion 5, nevíme o ničem pro Psion 3.

Zdroje informací

Psion HOWTO, *http://tldp.org/HOWTO/Psion-HOWTO.html.*
PLPtools, *http://plptools.sourceforge.net/,* je sada knihoven a nástrojů umožňující linuxovým systémům komunikovat s palmtopy Psion přes sériový port. Na Linuxu je možné i spojení přes IrDA s využitím funkce IrCOMM. Sdílená knihovna zapouzdřuje komunikační proto-kol vyšší vrstvy (PsionLinkProtocol), takže je možné vytvářet aplikace bez podrobnější zna-losti tohoto protokolu. Démon ncpd obsluhuje sériovou linku a zpřístupňuje ji na lokál-ním TCP soketu.
OpenPsion (dříve PsiLinux/Linux7k), *http://linux-7110.sourceforge.net/,* je port Linuxu na některé palmtopy.

Konektivita

Z linuxového počítače na nelinuxové PDA

Xcerdisp, *http://freshmeat.net/projects/xcerdisp/,* je ekvivalent Microsoft nástroje Remote DisplayControl určený pro prostředí X

Window. Čeká na spojení z cerdisp klienta běžícího pod WindowsCE na PocketPC a umožňuje PDA ovládat z linuxového počítače. Pro připojení PDA k síti budetemožná muset použít nástroj SynCE, *http://synce.sourceforge.net/synce/.*

Cílem projektu SynCE je zajistit komunikaci mezi Windows CE či Pocket PC a počítačem s Linu

xem, BSD nebo jiným unixovým systémem.Další informace o nástrojích pro zajištění propojení a synchronizace, o emulátorech a dalšíchnástrojích naleznete na adresách *http://tuxmobil.org/pda_linux.html* a *http://tuxmobil.org/howtos.html.*

Tabletová PC / Pen PC

Úvod

Tabletová PC jsou zvláštní typ notebooků. Typicky nemají klávesnici (nebo jsou vybaveny exter-ní či vzdálenou klávesnicí), mají dotykovou obrazovku (proto se jim také říká Pen PC) a obvykle bezdrátové připojení. V určitém smyslu je lze srovnávat s PDA. Microsoft má pro tabletová PC spe-ciální edici svého operačního systému a zveřejnil takzvanou specifikaci. V roce 2003 se začala pro-dávat první tabletová PC podle této specifikace, i když podobná zařízení s Linuxem už existovala mnohem dříve. Podívejte se na přehled linuxových notebooků s dotykovým displejem (*http:// tuxmobil.org/touch_laptops.html*), s odpojitelným displejem (*http://tuxmobil.org/detach_disp.html*) a také na přehled Linuxu na tabletových PC, WebPadech, Notepadech anebo Pen PC (*http:// tuxmobil.org/tablet_unix.html*). Tato zařízení se typicky používají ke sběru dat v obchodech, skla-dech a podobně, též jako čtečky knih a webové prohlížeče. Vzhledem ke svému hardwarovému řešení si žádají specifická linuxová řešení.

Displej

Dotykový displej

Dokument Linux Touch Screen HOWTO, *http://www.tldp.org/HOWTO/XFree86-Touch-Screen-HOWTO.html,* popisuje, jak nastavit X Window pro práci s dotykovou obrazovkou. K dispozici je také přehled linuxových notebooků s dotekovou obrazovkou a/nebo vstupem prostřednictvím pera (*http://tuxmobil.org/touch_laptops.html*) a přehled linuxu na tabletových PC (*http:// tuxmobil.org/tablet_unix.html*).

Otočení obrazovky X Window

Některé ovladače X.org/XFree86 podporují otočení obsahu obrazovky. V konfiguračním souboru zadejte požadovanou hodnotu – CW (90 stupňů po směru hodinových ručiček), CCW (90 stupňů proti směru hodinových ručiček) nebo UD (vzhůru nohama). Na různých ovladačích nemusí fungovat všechny možnosti:

```
Option "Rotate" "hodnota"
```

Od verze 4.3 obsahuje XFree86 rozšíření RandR (rozšíření pro změnu velikosti a otáčení), které umožňuje změnit rozlišení displeje za běhu, bez nutnosti restartovat X server. Nástroj xrandr pod-poruje pouze změnu rozlišení, nepodporuje otáčení. Server Tiny-X Keitha Packarda (jednoho z vývojářů RandR) implementuje všechny funkce. Novější X.Org také obsahují nástroj xrandr.

Nástroje

Pro linuxová PDA existuje několik nástrojů pro otočení obrazovky, zatím jsme je ale nezkoušeli. Hledejte na Zaurus Software Indexu – *http://killefiz.de/zaurus/.*

Rozpoznávání písma

Program xstroke, *http://davesource.com/Projects/xstroke/,* je určen pro rozpoznávání gest v systé-mu X Window. Zachycuje gesta generovaná ukazovacím zařízením (myší, stylusem, perem a podobně), rozeznává je a na jejich základě provádí akce. Program byl původně vyvíjen na linu-xových systémech (i386 a StrongARM), měl by však být snadno portovatelný na jakýkoliv unixo-vý systém používající X Window.

Program xscribble, *http://www.handhelds.org/projects/xscribble.html,* je aplikace, která uživatelům s dotykovými obrazovkami

umožňuje zadávat znaky pomocí „jednotahové“ abecedy. Pomocí roz-šíření X serveru umožňuje syntetizovat znaky stejně, jako kdyby byly zadávány na klávesnici. Byl určen pro Linux na PDA, stejným způsobem by však měl být použitelný i na tabletových PC.

Yudit, *http://www.yudit.org/,* je unicodový textový editor pro systém X Window. Dokáže zobra-zovat True Type fonty, tisknout, překódovávat vstup z klávesnice a rozpoznávat rukopis bez závis-losti na externích programech. Konverzní nástroje dokážou převádět text mezi různými kódová-ními. Jako konvertory textu lze použít i vstupní mapy klávesnice.

Klávesnice

Obrazovkové klávesnice xvkbd

xvkbd (*http://homepage3.nifty.com/tsato/xvkbd/*) je virtuální grafická klávesnice pro prostředí X Window. Umožňuje vkládat znaky do jiných klientských aplikací tak, že je vybíráte na „klávesni-ci“ zobrazené na obrazovce. Dokáže také jiným klientům poslat znaky předané na příkazovém řádku.

GNOME On-screen Keyboard (GOK)

Program GNOME On-screen Keyboard (GOK), *http://www.gok.ca/,* je dynamická obrazová klá-vesnice pro unixové operační systémy. Obsahuje funkce jako přímý výběr, automatické a inverz-ní skenování a podporuje automatické dokončování slov.

Vzdálená klávesnice

Některá tabletová PC jsou vybavena vzdálenou klávesnicí. Data mezi klávesnicí a počítačem se přenášejí infračerveným portem, přes BlueTooth nebo jiným podobným prostředkem. Pokud jde
o čistě hardwarové řešení, mělo by bez problémů fungovat i v Linuxu. V opačném případě budete pravděpodobně potřebovat technickou specifikaci od výrobce.

Virtuální klávesnice

Virtuální (fyzicky neexistující) klávesnice se řeší různými způsoby. Prozatím nemáme ověřeno, zda jednotlivá řešení fungují v Linuxu. Odkazy:

http://www.lumio.com/,
http://www.canesta.com/,
http://www.senseboard.com/,
http://www.lightglove.com/,
http://www.sait.samsung.co.kr/eng/main.jsp,
http://www.kittytech.com/.

Bezdrátové sítě

Viz kapitolu „Bezdrátové sítě – WLAN“.

Příklady

http://www.softwarekombinat.de/linux-point510.html.
http://libxg.free.fr/point/point.htm.
http://www.paceblade.com/site/desktopdefault.aspx.
http://opensimpad.org/index.php/Main_Page.

Na serveru TuxMobil naleznete informace o instalaci Linuxu na tabletová PC,
http://tuxmobil.org/tablet_unix.html.

Mobilní telefony, pagery

Na serveru TuxMobil naleznete informace o kompatibilitě různých mobilních telefonů a Linuxu (*http://tuxmobil.org/phones_linux.html*). Naleznete zde také odkazy na užitečné aplikace a na mobilní telefony, které na Linuxu přímo běží.

Mobilní telefony

Spojení s mobilními telefony bez linuxového operačního systému

Komunikaci s telefony Nokia řeší projekty GNOKII (*http://www.gnokii.org/*) a Nserver (*http://www.version6.net/misc/nserver.html*). Tento projekt usiluje o vytvoření GPL náhrady apli-kace Nokia Windows Nserver. Postupně se vyvíjí, v současné době umožňuje zálohovat a obno-vit data.

Projekt openWAP (*http://www.openwap.org/*) usiluje o implementaci protokolu WAP (Wireless Application Protocol), používaného v prohlížečích, serverech a různých nástrojích. Protokol WAP se používá k přenosu internetového obsahu na zařízení, jako jsou PDA, mobilní telefony, pagery a jiná bezdrátová zařízení.

Projekt GSMLIB (*http://www.pxh.de/fs/gsmlib/download/*) implementuje knihovnu pro přístup k mobilním telefonům GSM prostřednictvím GSM modemů. Knihovna nabízí mimo jiné funkce pro úpravy telefonního seznamu uloženého v telefonu nebo v SIM kartě, čtení, vytváření, odesílání a příjem SMS. Součástí projektu jsou i jednoduché nástroje, které umožňují tyto funkce používat.

Program Kannel (*http://www.kannel.org/*) je wapová brána. Usiluje o vytvoření volně dostupné otevřené implementace této základní části wapové infrastruktury, která umožní efektivní realizaci wapových služeb, ať už bezdrátovými operátory nebo specializovanými poskytovateli obsahu.

Kannel funguje také jako SMS brána pro GSM sítě. Příjem a odesílání SMS podporují prakticky všechny GSM telefony, takže je možno obsluhovat mnohem více klientů než jen uživatele nověj-ších telefonů s podporou wapu.

Mobilní telefony s Linuxem

Existuje několik mobilních telefonů, na nichž přímo běží Linux – *http://tuxmobil.org/-phones_linux.html.* Pro mobilní telefony existují také specializované linuxové distribuce – *http://tuxmobil.org/mobile_phone_linux_distributions.html.*

Pagery – SMS zprávy

QuickPage (*http://www.qpage.org/*) je softwarový balík obsahující klienta i server pro zasílání zpráv na alfanumerické pagery. Klient přijme od uživatele zprávu a protokolem SNPP ji předá ser-veru. Server pak zprávu odešle na pager příjemce prostřednictvím modemu a protokolu TAP (zná-mého též pod názvem IXO).

Program mail2sms (*http://daniel.haxx.se/projects/mail2sms/*) konvertuje e-mail (ve formátu MIME) na textovou zprávu. Podporuje vyhledávání a nahrazování, podmíněné operace, akce závislé na datu/čase, vlastní úpravy výstupního formátu a podobně. Standardní délka výstupu je 160 znaků, což je přesně to, co se dá v rámci SMS zprávy odeslat. Program samotný neobsahuje nic, co by provádělo vlastní odeslání zprávy, výsledek konverze je pouze vypisován na standard-ní výstup.

Program email2sms (*http://www.adamspiers.org/computing/email2sms/*) je filtr napsaný v Perlu, který konvertuje e-mail do podoby vhodné k odeslání prostřednictvím SMS. Jeho hlavní výhodou oproti předešlému programu je to, že pomocí CPAN modulu *Lingua::EN::Squeeze* komprimuje text přibližně na 40 % původní velikosti, takže do 160znakového limitu délky SMS zprávy „natlačíte“ mnohem větší část e-mailu. Filtr je plně kompatibilní s MIME a obsahuje mnoho nastavitelných funkcí, například odstranění citované části textu. Výborně se používá v kombinaci s procmailem. Obsahuje rovněž skript pro odeslání zprávy prostřednictvím typické webové brány.

Program SMSLink (*http://smslink.sourceforge.net/*) implementuje SMS bránu s architekturou kli-ent/server. Vyžaduje použití specializovaného hardwaru (sériového GSM modulu). Podporuje pří-jem i odesílání SMS. Server v současné době běží jen pod Linuxem a prostřednictvím služby tel-net podporuje i interaktivní režim. Řádkový klient je dostupný pro Linux, Solaris a HP-UX. Nabí-zí také základní webové rozhraní. Pracuje se na klientovi pro Win32.

Program nmsms (*http://fresh.t-systems-sfr.com/unix/src/privat2/nmsms-0.05.tgz/*) je velmi jedno-duchý program oznamující příchozí a e-mail na SMS adresu (e-mailovou adresu) zadanou při pře-kladu. Do oznámení se přenáší obsah hlaviček *From:* a *Subject:* z původního e-mailu.

Program mepl (*http://www.hof-berlin.de/mepl/en.html*) je určen k řízení režimu „self-employed“ messagemodemů 3COM/USRobotics. Program lze použít ke stažení zpráv a jejich uložení nebo zaslání ve formátu GSM nebo faxu.

Kalkulačky,

digitálnífotoaparáty, doplňky

I když jde o téma, které podle našeho názoru se zaměřením tohoto dokumentu souvisí, prozatím se mu příliš nevěnujeme. Obecné informace o těchto systémech naleznete na serveru Embed-ded.com (*http://www.embedded.com/*). Informace související s Linuxem nabízejí projekty ELKS (*http://elks.sourceforge.net/*) a uCLinux (*http://uclinux.org/*). Zajímavá je také diskusní skupina comp.arch.embedded.

Digitální fotoaparáty

Související dokumentace

Kodak Digital Camera HOWTO Davida Burleyho, *http://www.marblehorse.org/projects/documen-tation/kodak/*.

Úvod

Informace o mobilních telefonech a digitálních fotoaparátech naleznete v dokumentech *http:// tuxmobil.org/ir_misc.html* a *http://tuxmobil.org/howtos.html.* Dalším zdrojem informací je diskusní skupina rec.photo.digital.

Adaptér Flashpath je disketě podobné zařízení, které slouží k přenosu dat z digitálního fotoapa-rátu na počítač. Podívejte se na projekt Flashpath for Linux (*http://www.smartdisk.com/ Downloads/FPDrivers/LinuxDownload.htm*) a na stránky Jamese Radleyho (*http://www. susie.demon.co.uk/flashpath.html*). Poznámka: Nejedná se o oficiálně certifikované zařízení a není uvolněno pod GPL.

Kalkulačky

Informace o kalkulačkách jako HP-48 naleznete jednak na serveru HP-Calculator.Org (*http://www.hpcalc.org/*) a také na Keithových stránkách o HP-48 (*http://www.gmi.edu/ ~madd0118/hp48/*). Dokument „HP-48 Kermit Hints and Tips“ (*http://www.columbia.edu/ kermit/hp48.html*) popisuje, jak s kalkulátorem HP-48 komunikovat prostřednictvím jeho sériové-ho portu protokolem Kermit. Kalkulačku HP-48 lze také použít jako linuxový terminál (*http://panic.et.tudelft.nl/~costar/hp48*).

Další informace naleznete také na stránkách *http://tuxmobil.org/ir_misc.html.*
Existuje zálohovací nástroj pro diáře Casio, *http://www.tunbury.demon.co.uk/casio/,* jde o port původního dosového programu, který umožňuje komunikaci s organizátory Casio. Umožňuje zálohovat data z organizátoru do počítače a obnovit data v organizátoru ze zálohy. Dovoluje také vypsat data z organizátoru v čitelné podobě. V současné době podporuje manipulaci s telefonem, kalendářem, plánovačem a poznámkovým blokem. S tématem souvisejí také strán-ky Alank (*http://www.aloha.net/alank/*), Casio World (*http://www.casioworld.com/*), SunSite Archiv (*http://www.ibiblio.org/pub/Linux/apps/*) a stránky Milana Urosevice (*http://home. t-online.de/home/Milan.Urosevic/*).

Program GtkTiLink (*http://membres.lycos.fr/rlievin/*) umožňuje přenos dat mezi kalkulátory Texas Instruments a počítačem. Podporuje všechny typy kabelů (paralelní, sériový, Black and Gray TI Graph Link). Umí pracovat s kalkulátory TI82, TI89, TI92 a TI92+. Dokáže přijímat a odesílat data a zálohy, snímat obrazovku kalkulátoru a vzdáleně kalkulátor ovládat.

Wearables

S tématy Linuxu a mobilních počítačových zařízení souvisí také wearable zařízení. Podívejte se na stránky MIT (*http://www.media.mit.edu/wearables/*), Wearables Central (*http://wearables.blu.org/*) a WearComp (*http://www.wearcomp.org/*).

Vzhledem k problémům s provozováním klasického grafického uživatelského prostředí na wea-rable počítačích vznikl projekt Sulawesi (*http://tldp.org/HOWTO/Wearable-HOWTO-7.html*). Byl navržen a implementován tak, aby řešil všechny důležité problémy wearable rozhraní – tedy mož-nost přijímat vstupy z mnoha různých zdrojů, například prostřednictvím strojového vidění, rozpo-znávání řeči, přenosné klávesnice, GPS zařízení, infračervených zařízení, také možnost a genero-vat vhodné výstupy, například prostřednictvím generování řeči, náhlavních displejů, vibračních signalizací a podobně. Postupem vývoje došlo k aktualizaci uživatelského rozhraní *Gili*, rozšířila se dokumentace a byl uveden *Spatial Reminder*.

Hodinky

Knihovna datalink (*http://datalink.fries.net/*) umožňuje posílat informace na hodinky Timex Data-Line. Původní knihovna

podporuje modely DataLink 70, 150 a 150S, později byla rozšířena o pod-poru modelu DataLink Ironman Triathlon. Byla testována podpora SVGA na hodinkách Ironman, jiná výstupní zařízení a jiné hodinky mohou a nemusí fungovat. Displej musí být CRT (nikoliv LCD).

Portable Play Station

Program qpspmanager (*http://qpspmanager.sourceforge.net/*) umožňuje spravovat soubory na memorysticku, používaném v Sony Playstation Portable.

Podrobnosti o hardwaru

– CPU, displej, klávesnice a další

Úvod

Následující text se podrobněji věnuje hardwaru mobilních zařízení, platí pro všechna mobilní zaří-zení, na nichž lze použít Linux – notebooky, PDA, HPC, mobilní telefony a podobně, samozřej-mě občas s drobnými modifikacemi.

BIOS

Než začnete jakékoliv hardwarové zařízení nějak nastavovat, podívejte se nejprve do BIOSu. Často zde naleznete řešení problému – například možnost nastavení displeje, zapnutí APCI či APM, nastavení DMA, IrDA, PCMCIA, zvuku a podobně.

Pokud při nastavování hardwaru narazíte na nepřekonatelné potíže, můžete vyzkoušet upgrado-vat BIOS novější verzí od výrobce. K této operaci budete pravděpodobně potřebovat některýz operačních systémů společnosti Microsoft nebo alespoň disketu s DOSem.

Aktualizace BIOSu se mění ve stále větší problém, protože jak DOS, tak disketové mechanikypostupně vymírají. Pokud navíc používáte výhradně GNU/Linux, je to ještě složitější. Naštěstímůžete vytvořit bootovatelné CD s GNU/Linuxem, které vám umožňují dosovým nástrojem aktu-alizovat BIOS, bez nutnosti použít Windows, MS-DOS nebo disketovou mechaniku. Příslušný pro-jekt naleznete na adrese *http://freshrpms.net/docs/bios-flash/*.

Některé novější notebooky, například ASUS M5200A, mají BIOS, který dokáže aktualizovat přímo

sám sebe.Dokument Motherboard Flash Boot CD from Linux HOWTO, *http://www.nenie.org/misc/flashbootcd.html,* uvádí návod, jak v Linuxu (nebo jiném Unixu) vytvořit bootovací disk pro pře-flashování BIOSu v situaci, kdy nemáte k dispozici ani disketovou mechaniku, ani přístup k MS-DOSu či Windows.

Projekt LinuxBIOS, *http://www.linuxbios.org,* představuje náhradu klasického BIOSu z PC, Alphya jiných strojů linuxovým jádrem, které dokáže nabootovat Linux přímo ze studeného startu systé-
mu. LinuxBIOS je primárně Linux – jde o asi desetiřádkový patch jádra. Kromě toho obsahuje spouštěcí kód (asi 500 řádků assembleru a 5 000 řádků C), který provede 16 instrukcí pro pře-pnutí procesoru do 32bitového režimu a následně otestuje paměť a provede inicializaci hardwaru do stavu, ve kterém už může běžet Linux. Máme zprávy o úspěšném použití LinuxBIOS na note-booku.

SMBios

Specifikace Desktop Management Interface (DMI) Standards, *http://www.dmtf.org/ standards/dmi/,* vytváří standardní framework pro správu a sledování komponent ve stolním počí-tači, notebooku či serveru. DMI byl první standard pro správu počítače. Domovská stránka DMI obsahuje všechny související informace, od specifikací po nástroje pro registraci produktů s DMI certifikací.

Program dmidecode, *http://www.nongnu.org/dmidecode/,* zobrazuje informace o hardwaru syste-mu tak, jak je popisuje BIOS ve shodě se standardem SMBIOS/DMI. Mezi uváděné informace typicky patří výrobce systému, název modelu, sériové číslo, verze BIOSu a celá řada dalších podrobností, podle konkrétního výrobce více či méně zajímavých a více či méně spolehlivých. Velmi často jsou například poskytovány informace o využití procesorových soketů, rozšiřujících slotů (AGP, PCI, ISA), paměťových slotů a seznamy vstupně-výstupních portů (sériových, paralel-ních, USB).

Existuje ještě alternativní implementace dekodéru DMI tabulky. Knihovna libsmbios, *http://linux.dell.com/libsmbios/main/index.html,* je víceplatformní knihovna, jejímž smyslem je poskytnout unifikované API pro získání typických informací dostupných v BIOSu. V současné době dokáže přistupovat ke všem informacím v tabulkách SMBIOS. Dokáže také získat specifické informace systémů Dell, například ID systému nebo servisní záznam. K některým často používa-ným funkcím existuje API v jazyce C a příklady programů, které ukazují práci s knihovnou.

CPU

Přehled procesorů používaných v mobilních zařízeních a podporovaných Linuxem naleznete v úvodní kapitole tohoto dokumentu.

SpeedStep

SpeedStep je funkce novějších procesorů společnosti Intel, která umožňuje měnit frekvenci pro-cesoru. Pro Linux existuje několik nástrojů, které dokážou s tímto mechanismem pracovat. Podob-né možnosti nabízejí i procesory AMD a StrongARM.

Než začnete SpeedStep nastavovat, podívejte se na volby BIOSu.

Nástroj SpeedStep

Nástroj SpeedStep, *http://www.goof.com/pcg/marc/speedstep.html,* spolupracuje pouze s procesory Mobile Pentium-III. Podívejte se na výpis příkazu cat /proc/cpuinfo:

```
model name : Intel(R) Pentium(R) III Mobile CPU 1000MHz
```

S klasickými procesory nespolupracuje:

```
model name : Pentium III (Coppermine)
```

CPUFREQ

Možná vás bude zajímat patch cpufreq, *http://www.brodo.de/cpufreq,* pro jádra 2.4/2.5 (od 2.6 je součástí jádra). Podporuje změnu frekvence procesorů x86 a ARM. Modul poskytuje rozhraní pro uživatelský prostor a standardní jaderné rozhraní. Spotřeba procesoru je přímo spjata s jeho frek-vencí, takže provozování procesoru na nejnižší dostačující frekvenci vede k prodloužení doby pro-vozu z baterií. Celý mechanismus lze používat k dynamické optimalizaci výkonu (frekvence) ver-sus spotřeby mnohokrát za sekundu.

cpufreqd

Démon cpufreqd, *http://sourceforge.net/projects/cpufreqd,* funguje podobně jako applety pro nastavování frekvence procesoru, které znáte z jiných operačních systémů. Sleduje úroveň nabití baterií, stav síťového napájení a spuštěné programy a upravuje frekvenci procesoru podle pravi-del definovaných v konfiguračním souboru. Dokáže pracovat s APM i s ACPI.

cpudyn

Program cpudyn, *http://mnm.uib.es/gallir/cpudyn/,* řídí rychlost procesorů Intel SpeedStep a PowerPC s podporou cupfreq přeloženou v jádře. Šetří baterie a snižuje teplotu, aniž by ovliv-ňoval výkon interaktivních aplikací.

cpuspeedy

Program cpuspeedy, *http://cpuspeedy.sourceforge.net/,* umožňuje měnit frekvenci hodin a napá-jení procesoru prostřednictvím linuxového ovladače CPUFreq. Jde o program běžící v uživatel-ském prostoru a podporuje tak všechny procesory podporované ovladačem CPUFreq.

powernowd

Program PowerNowd, *http://www.deater.net/john/powernowd.html,* je jednoduchý klientský démon ovladače cpufreq, který používá rozhraní sysfs. Běží na pozadí a mění frekvenci proceso-ru v konfigurovatelných krocích podle aktuálního využití. Je

napsán v jazyce C, chlubí se svou rychlostí a jednoduchostí. Má mnoho konfiguračních možností, podporuje i jiné procesory než x86 a podporuje rovněž SMP.

Režim notebooku

Režim notebooku, *laptop mode*, *http://www.samwel.tk/laptop_mode/,* je „režim" jádra, který pro-dlužuje výdrž baterií. Dosahuje toho inteligentním shlukováním diskových zápisů, takže k vynu-cenému roztočení disku dojde pouze při čtení dat nenacházejících se ve vyrovnávací paměti. Údaj-ně dosahuje významného prodloužení doby běhu na baterie, samozřejmě jen při vhodných „obrazcích užití" notebooku.

Balík Laptop Mode Tools, *http://www.samwel.tk/laptop_mode/tools/index.html,* zastavuje pevný disk podobně jako noflushd, funguje však i se žurnálovacími souborovými systémy. Má integro-ránu podporu apmd/acpid/pbbuttonsd, takže se zmíněné chování zapíná jen tehdy, když note-book běží na baterie. Kromě toho upravuje některé hodnoty pro hdparm, připojuje souborové systémy s příznakem noatime a umožňuje nastavit maximální frekvenci procesoru.

Démon Sony VAIO SPIC

Démon Sony VAIO SPIC, *http://spicd.raszi.hu/,* je drobná úprava, která umožňuje použití apmd na noteboocích Sony VAIO. Prostřednictvím jaderného modulu sonypi detekuje stav síťového napájení a podsvícení displeje, pomocí cpufreqd řídí frekvenci procesoru.

CPUIDLE

Jde o softwarový nástroj, *http://www.cpuidle.de/index.php,* který snižuje teplotu procesoru. Přijde vám to divné? Je to prosté – přemýšleli jste někdy nad tím, že po většinu času je procesor vaše-ho počítače nečinný? Pokud například používáte textový editor, píšete e-mail nebo prohlížíte webové stránky, procesor po většinu času nedělá vůbec nic, pouze čeká na vstup. Při tomto nic-nedělání ovšem spotřebovává třeba 30 W a vytváří nezanedbatelné množství tepla. Rozumné ope-rační systémy jako Linux, NT nebo OS/2 mají takzvanou *idle loop* – smyčku, která se vykonává vždy, když procesor nemá co dělat. Tato smyčka je tvořena instrukcemi HLT. Procesory jako AMD K6, Cyrix 6x86 a 6x86MX mají speciální funkci nazvanou „suspend-on-halt". Znamená to, že kdy-koliv procesor vykonává instrukci HLT, přechází na krátkou dobu do režimu pozastavení. V těle této smyčky je tak procesor „uspán", spotřebovává méně energie a produkuje méně tepla. Samo-zřejmě to nijak neovlivňuje výkon! Uživatel si ani nevšimne, že po většinu času jeho „práce" pro-cesor vlastně spí – přesvědčí se o tom pouze sáhnutím na chladič.

autospeedstep

Démon autospeedstep, *http://gpsdrive.kraftvoll.at/speedstep.shtml,* řídí spotřebu energie a frekven-ci procesoru v závislosti na jeho zatížení. Pracuje s procesory Intel SpeedStep a s jádry s podpo-rou ACPI.

ACPI

Pokud používáte jádro s podporou ACPI, můžete parametry funkce SpeedStep nastavovat i přes rozhraní /proc/apci/, například příkazem echo 1 > /proc/acpi/processor/CPU0/performan-ce procesor zpomalíte. Poznámka: Mezery v příkazu jsou důležité. Můžete také použít následující skript Sebastiana Henschela:

```
#! /bin/sh

# /etc/init.d/slowcpu: slow down cpu or accelerate it via speedstep

test -e /proc/acpi/processor/CPU0/performance || exit 0

case "$1" in
        start)
        echo "Setting CPU0-Speed to: 733 MHz."
        echo 1 > /proc/acpi/processor/CPU0/performance
                ;;
        stop)
        echo "Setting CPU0-Speed to: 1133 MHz."
        echo 0 > /proc/acpi/processor/CPU0/performance
                ;;
        force-reload|restart)
                ;;
        *)
                echo "Usage: $0 {start|stop}"
                exit 1 esac

exit 0
```

Další poznámka: Pro jádra >2.6.11 je tato funkce zastaralá a je nutno použít /sys, například echo 800000 > /sys/devices/system/cpu/cpu0/cpufreq/scaling_max_freq (podporované frekven-ce zjistíte v souboru

/sys/devices/system/cpu/cpu0/cpufreq/scaling_available_frequ-encies). Úprava výše uvedeného skriptu tak, aby používal /sys, je pak triviální.

Centrino

Technologie Centrino(TM) je tvořena trojicí komponent: procesorem Pentium-M, čipsetem a bez-drátovým modulem. Podívejme se, jak jsou jednotlivé komponenty momentálně v Linuxu podpo-rovány. Oficiální stanovisko Intelu k podpoře Linuxu naleznete na jejich stránkách věnovaných kompatibilitě s jednotlivými operačními systémy, *http://www.intel.com/support/chipsets/.*

Na stránce *http://tuxmobil.org/centrino.html* naleznete informace o Linuxu na noteboocích s tech-nologií Centrino.

CPU: Pentium-M

Podle informací na stránkách Intelu je rodina procesorů Pentium-M v Linuxu podporována. Robert Freund napsal stručný dokument o softwarovém řízení ACPI funkcí Centrina v Linuxu. Dokument naleznete na adrese *http://rffr.de/acpi,* popisuje se v něm, jak ovládat frekvenci procesoru a další režimy šetření spotřeby i jak zjistit informace o stavu baterie.

Čipset 855/915

Rodiny čipsetů Intel 855/915 jsou navrženy tak, aby poskytovaly lepší výkon při nižší spotřebě. Čipsety se vyrábějí buď jako samostatný řadič paměti (Intel 855PM) nebo jako integrovaný řadič paměti a grafický čip (Intel 855GM). Pro Linux nabízí Intel ovladač Extreme Graphics, který obsa-huje binární soubory jaderných ovladačů AGP GART a DRM. S těmito ovladači nemáme žádné zkušenosti, protože čipset funguje i se standardními ovladači z XFree86/X.org. Procesor Pentium--M může být dodáván i v kombinaci s jinými grafickými čipsety, například od ATI, nVidie či Tri-dentu.

Bezdrátová sí – MiniPCI adaptér PRO/Wireless 2100/2200

Existuje několik řešení, jak tyto karty provozovat v Linuxu: nativní ovladače pro Linux, použití NDIS wrapperu nebo komerční Linuxant driverloader. Ovladač *http://ipw2100.sourceforge.net/* je originální ovladač s otevřeným kódem přímo od Inte-lu, zahrnuje i potřebný firmware a je určen pro první generaci Centrin (včetně WEP a WPA spolu s HostAP). Ovladače pro druhou generaci miniPCI modulů PRO/Wireless 2200BG (802.11g/802.11i) poskytuje projekt *http://ipw2200.sourceforge.net/.* Tento projekt podporuje i třetí generaci karet, PRO/Wireless 2915ABG (IEEE 802.11b, 802.11g a 802.11a).

V době, kdy Intel uvedl technologii Centrino na trh, ještě neposkytoval linuxové ovladače. Proto vznikala jiná řešení – různí výrobci bezdrátových čipů odmítali zveřejnit technické specifikace nebo alespoň binární linuxové ovladače. Tuto situaci řešil NDIS wrapper, který poskytuje jaderný modul, jímž je možné zavést ovladače podle specifikace NDIS (Microsoft Windows Network Dri-ver Interface Specification). V současné době existují dvě implementace. Komerční Linuxant Driverloa-der, *http://www.linuxant.com/driverloader/,* podporující celou řadu čipsetů včetně Pro/Wireless 2100 od Intelu. Druhá varianta je volně dostupný ndiswrapper, *http://ndiswrapper.sourceforge.net/,* Pontuse Fuchse.

Dalším řešením samozřejmě je používat bezdrátové karty, které Linux podporuje. Jejich seznam naleznete na stránce *http://tuxmobil.org/minipci_linux.html.* Tyto karty se shánějí obtížně, nicmé-ně miniPCI karty můžete nalézt v některých klasických PCI kartách pro stolní PC. Vypreparovat takovou kartu a použít ji v notebooku může být poměrně obtížný proces. Manuál k tomuto postu-pu napsal Theodore Tytso, jeden z vývojářů jádra. Naleznete jej na následující adrese: *http://www.thunk.org/tytso/linux/t40.html.* Můžete také použít bezdrátovou kartu PCMCIA nebo CF. Takové řešení může být pružnější, protože PCMCIA či CF kartu můžete používat ve více zaří-zeních a podle potřeby si k nim vyberete linuxový ovladač. K některým kartám je možné připojit i anténu a zvětšit tak dosah bezdrátového zařízení. Seznam těchto karet kompatibilních s Linuxem naleznete na adrese *http://tuxmobil.org/pcmcia_linux.html.* Lze očekávat, že výrobci v budoucnu nabídnou alternativní řešení na bázi miniPCI, například DELL už takové řešení nabízí v sérii Lati-tude D.

Shrnutí

I když podpora v Linuxu není prozatím úplná, některé vlastnosti technologie Centrino už stojí vzít v úvahu při výběru příštího notebooku. I když jsou názvy těchto nových procesorů voleny podob-ně ke stávajícím, a lidé si je proto často pletou, uvnitř jde o úplně jiné čipy. V porovnání s pro-cesorem Pentium-4 Mobile je Pentium-M menší a umožňuje tak výrobu lehčích a kompaktnějších přenosných zařízení. Vzhledem k vysoké taktovací frekvenci produkují procesory Pentium-4 pří-liš mnoho tepla, než aby je bylo možné vestavět do úzkých skříní notebooků. Extrémně ploché notebooky se proto doposud vyráběly jen s procesory Pentium III Mobile. Při stejném výkonu má navíc Pentium-M menší spotřebu z baterií, nemáme však přesně změřeno, o jak velkou úsporu se jedná. PENN Computing na adrese *http://www.upenn.edu/computing/provider/docs/centrinoprovi-der.html* nabízí srovnání procesorů Pentium-M a Pentium-4 Mobile. Poznámka: Písmeno „M“ v názvu Pentium-M naznačuje slovo „mobile“. Proto si řada lidí plete tyto procesory s mobilní verzí procesorů Pentium-III a Pentium-4.

Notebooky založené na technologii Centrino jsou už dnes v linuxové komunitě velmi oblíbené. Na serveru TuxMobil, *http://tuxmobil.org/centrino.html,* naleznete zprávy o instalaci Linuxu prak-ticky na všechny existující notebooky s technologií Centrino.

Řadiče PCMCIA

Test kompatibility s Linuxem

Příkazem probe, který je součástí balíku PCMCIA-CS Davida Hindse, můžete zjistit typ řadiče PCMCIA. Stejné informace zjistíte příkazem cat /proc/pci.

Související dokumentace

PCMCIA HOWTO, *http://pcmcia-cs.sourceforge.net/ftp/doc/PCMCIA-HOWTO.html*

Konfigurace PCMCIA

V poštovní konferenci, jejímž jsem členem, se čas od času objeví dotaz: „Jak můžu po instalaci Linuxu nastavit podporu PCMCIA?“ Proto uvádíme následující krátký návod. Autoritativním zdro-jem nejnovějších informací o podpoře PCMCIA, včetně dokumentace, souborů ke stažení a obec-ných informací o PCMCIA, jsou stránky *http://pcmcia-cs.sourceforge.net/.* O problémech mezi PCMCIA a APM hovoříme v kapitole o APM.

Software

Nainstalujte si nejnovější balíček PCMCIA-CS, pokud použijete balík rpm či deb pro vaši distribuci, je to velmi snadné.
Přečtěte si dokument PCMCIA HOWTO, typicky je součástí balíčku PCMCIA-CS.
Je-li to nutné, nainstalujte nové jádro.
Ověřte, že máte jádro přeloženo s podporou modulů a PCMCIA (a většinou také s podporou APM).
Ověřte, že máte jádro přeloženo s podporou toho, co zamýšlíte použít – například tedy s podporou sítě u síťových karet, podporou sériových portů u modemových karet, SCSI u SCSI karet a podobně.
Máte-li vlastní jádro, nezapomeňte přeložit balík PCMCIA-CS proti svému jádru.

Řadič PCMCIA

Příkazem probe zjistíte, zda systém nadetekoval váš řadič PCMCIA.
Upravte soubor /etc/sysconfig/pcmcia. Měl by obsahovat záznamy PCMCIA=y a typ PCMCIA řadiče, například PCIC=i82365.
Počínaje jádry 2.6 existuje standardní ovladač PCIC=yenta_socket.
Spusťte službu PCMCIA, typicky příkazem /etc/init.d/pcmcia start. Ozvou-li se dvě píp-nutí, mělo by být všechno v pořádku.
Pokud něco nefunguje, zkontrolujte zprávy v souboru /var/log/messages.

PCMCIA karty

Ověřte přítomnost karty příkazem cardctl ident.
Pokud v souboru /etc/pcmcia/config není vaše karta uvedena, vhodně modifikujte sou-bor /etc/pcmcia/<KARTA>.conf. Řiďte se podle obsahu prvního zmiňovaného souboru. Můžete vyzkoušet různé ovladače, třeba karta začne fungovat – například ovladač pcnet_cs podporuje celou řadu síťových karet PCMCIA kompatibilních s NE2000. Po-známka: Není rozumné editovat přímo soubor /etc/pcmcia/config, protože o všechny provedené změny při příští aktualizaci přijdete.
Součástí balíku PCMCIA-CS je i seznam podporovaných karet. Aktuální verzi tohoto sezna-mu naleznete na adrese *http://pcmcia-cs.sourceforge.net/ftp/SUPPORTED.CARDS.*
Pokud používáte grafické rozhraní X Window, můžete použít příkaz cardinfo a přidávat, pozastavovat nebo restartovat PCMCIA karty v pěkném grafickém rozhraní.

Grafické čipy

Ověření kompatibility s Linuxem Grafický režim

Upozornění: Nástroj SuperProbe je zastaralý. SuperProbe je součástí balíku XFree86 (v X.org již není) a dokáže detekovat celou řadu grafických čipů. Přečtěte si pozorně dokumentaci, protože použití tohoto nástroje může vést k pádu počítače. Z manuálové stránky programu:

„SuperProbe je program, který se pokouší zjistit typ videohardwaru připojeného do sběrnice EISA/ISA/VLB – pokouší se detekovat přítomnost různých kombinací známých registrů na různých místech (sběrnice MicroChannel a PCI nejsou plně podporovány, řada z nich ale funguje při pou-žití volby -no_bios). Jedná se o proces náchylný k chybám, zejména na unixových systémech (které obvykle používají mnohem esoteričtější hardware než systémy MS-DOS), proto bude SuperProne pravděpodobně potřebovat pomoc uživatele.

V současné době SuperProne dokáže detekovat karty MDA, Hercules, CGA, MCGA, EGA, VGA a celou řadu čipsetů SVGA (viz volba -info). Dokáže identifikovat také některé HiColor/TrueCo-lor RAMDAC na SVGA kartách a u řady karet dokáže zjistit velikost instalované paměti. Dokáže identifikovat 8514/A a některé odvozeniny, nedokáže ale detekovat XGA a PGC. Nedokáže také identifikovat některé esoterické videokarty jako Targa, TIGA nebo Microfield.“

Pro účely testování spusťte server X11 příkazem X 2> <chybový.soubor>. Klávesovými kombi-nacemi Ctrl+Alt+Num+ a Ctrl +Alt+Num-vyzkoušejte změnit rozlišení. Poznámka: Klávesy Num+ a Num-je + a -na numerické klávesnici, na některých noteboocích ji musíte emulovat pomocí klávesy Fn.

Textový režim

Podívejte se na displej a uvidíte, zda funguje správně. Pokud ne, zkuste nabootovat s jiným video-režimem. Nalezení správného nastavení může představovat pokročilé cvičení v metodě pokus-omyl.

Související dokumentace

Nejprve si přečtěte dokumentaci k samotnému systému XFree86, *http://www.xfree86.org/.* Většinou je nainstalována lokálně jako /usr/share/doc/xfree86*. Samozřejmě, pokud používáte X.Org, zvolte dokumentaci k tomuto systému, *http://x.org/.*
XFree86 HOWTO, *http://tldp.org/HOWTO/XFree86-HOWTO/.*
XFree86 Video Timings HOWTO, *http://tldp.org/HOWTO/XFree86-Video-Timings-HOWTO/.*
XFree86 XInside HOWTO, *http://tldp.org/HOWTO/XFree86-XInside.html.*
X Big Cursor miniHOWTO, *http://tldp.org/HOWTO/X-Big-Cursor.html* (užitečné, pokud X11 používáte na notebooku s málo kontrastním LCD).
Keyboard and Console HOWTO, *http://tldp.org/HOWTO/Keyboard-and-Console-HOWTO.html.*
Framebuffer HOWTO, *http://tldp.org/HOWTO/Framebuffer-HOWTO.html.*

X servery

Může se vám stát, že některé funkce vašeho notebooku nebudou servery XFree86 ani X.Org pod-porovat – může jít například o vyšší rozlišení, akceleraci nebo externí monitor. Proto zde uvádí-me přehled i některých dalších X serverů.

XFree86, *http://www.xfree86.org/*
X.Org, *http://x.org/*
Zařízení VESA Framebuffer, dostupné v jádru 2.2.x, XFree86 3.3.2 a vyšších. Přečtěte si dokumenty *http://linux-fbdev.org/* a *http://linux.bytesex.org/* a dokumentace ve zdrojovém stromu jádra v /usr/src/linux/Documentation. Pokud používáte dedikovaný ovladač framebufferu pro čipy NeoMagic a jiné čipsety s podporou akcelerace, vždy používejte nej-novější verzi knihovny DirectFB, *http://linux.bytesex.org/.* DirectFB je malá knihovna, která programátorům umožňuje využívat hardwarovou akceleraci, abstraktní vrstvu pro práci se vstupními zařízeními a integrovaný okenní systém s podporou více zobrazovaných vrstev a průhledných oken, to vše nad framebufferovým zařízením. Jde o úplnou hardwarovou abstrakční vrstvu se softwarovým fallbackem u všech grafických operací, které hardware přímo nepodporuje.
Xi Graphics, *http://www.xig.com/,* komerční server známý též pod staršími názvy Accele-ratedX nebo Xinside.
SciTech, *http://www.scitechsoft.com/,* komerční server.
Metro-X, *http://www.metrolink.com/,* komerční server.

Pokud se vám nepodaří zprovoznit některý z běžných volně dostupných X serverů a zároveň nechcete utrácet peníze za komerční server, můžete vyzkoušet servery VGA16 nebo mono, které jsou součástí distribuce XFree86/X.org.

Zdroje informací

Na serveru Tuxmobil na adrese *http://tuxmobil.org/graphic_linux.html* naleznete přehled grafic-kých čipů, které se dnes v noteboocích používají.

Externí monitory: LCD, CRT, TV, projektor

Existuje několik způsobů aktivace podpory externího monitoru: například pomocí voleb BIOSu
nebo za běhu stiskem vhodné klávesové kombinace, například <Fn><F4>. Pozorně si přečtěte dokumentaci X serveru týkající se vašeho grafického čipu – například u čipů NeoMagic NM20xx je nutné upravit soubor /etc/XF86Config a nastavit volby intern_disp a extern_disp. Poznámka: Pokud víme, tyto volby fungují jen v XFree86 3.3.x, v XFree86 4.x ani X.org žádná podobná volba není.

Pokud se vám s XFree86/X.org nepodaří zprovoznit externí monitor, vyzkoušejte demoverzi některého z výše uvedených komerčních serverů. Podívejte se také na stránky distribucí RedHat či SUSE, které občas nabízejí nové, jen binární X servery, které mohou podporovat některé speciální funkce různých notebooků.

Nástroje

Nástrojem atitvout, *http://0pointer.de/lennart/projects/atitvout/,* můžete v GNU/Linuxu zadávat konfigurační příkazy pro konektor TV Out na kartách ATI Rage Mobility P/M. Primární slouží k zapnutí podpory TV Out po nabootování a k přepínání televizní normy mezi NTSC a PAL.
Programem s3switch, *http://www.probo.com/timr/savage40.html,* můžete výstup přepínat mezi různými zařízeními podporovanými kartami Savage (CRT, LCD, TV).Program nv-tv-out, *http://sourceforge.net/projects/nv-tv-out,* slouží k zapnutí televizního výstupuna kartách NVidia. Nevyžaduje podporu v jádře a podporuje několik různých televizních čipů.Můžete využívat všech služeb čipu až na úroveň přímého přístupu k jeho registrům a všechna roz-lišení a velikosti, které čip podporuje.

i810switch, *http://www16.plala.or.jp/mano-a-mano/i810switch.html,* je nástroj pro přepínánímezi LCD a externím VGA displejem prakticky na všech grafických čipech rodiny Intel i8xx, včet-ně technologie Centrino.

i855crt, *http://sourceforge.net/projects/i855crt,* je program běžící v uživatelském prostoru, kterýmje možné zapnout výstup pro externí monitor na noteboocích s čipsetem Intel 855GM.

Řešení

Klaus Weidner popsal konfiguraci se dvěma monitory bez použití programu xinerama, místo něj používá x2vnc. Díky tomuto řešení je možné druhý monitor dynamicky přidávat a odebírat bez nutnosti cokoliv překonfigurovávat nebo restartovat. Popis této konfigurace naleznete na adrese
http://mailman.linux-thinkpad.org/pipermail/linux-thinkpad/2003-November/013701.html.

Různé

Občas se může stát, že displej nefunguje správně v textovém režimu. Pro takovou situaci nemáme žádné doporučení, podívejte se do dokumentu Keyboard and Console HOWTO,
http://tldp.org/HOWTO/Keyboard-and-Console-HOWTO.html.

Věnujte pozornost podsvícení displeje, tato zařízení mají obvykle jen omezený počet cyklů zapnutí/vypnutí. Vyhýbejte se proto častému používání spořičů obrazovky.
O problémech spolupráce X Window a APM hovoříme v kapitole o APM. Nástroj vbetool, *http://www.srcf.ucam.org/~mjg59/vbetool/,* spouští prostřednictvím LRMI kód z video BIOSu. V současné verzi tak dokáže přepínat mezi DPMS stavy, uložit a obnovit stav videokarty a provést inicializaci videokarty. Primárně je určen ke zvýšení pravděpodobnosti úspěš-ného obnovení stavu videokarty po operaci ACPI S3 suspent-to-RAM.

DVI port

Pokud víme, v současné době DVI porty v Linuxu nefungují. Pro jistotu ale uvádíme odkaz na dokumenty popisující instalaci Linuxu na noteboocích s DVI portem - *http://tuxmobil.org/ laptop_dvi_linux.html.*

Videoport / ZV port

Některé notebooky vyšší třídy jsou vybaveny videoportem či ZV portem (NTSC/PAL). S tako-vými notebooky nemáme přímé zkušenosti, proto vám nabízíme pouze odkazy na ovladač BTTV (*http://www.metzlerbros.org/*) a program xwintv (*http://www.mathematik. unikl.de/~wenk/xwintv.html*). Další informace můžete nalézt na serveru Video4Linux, *http://www.video4linux.net/.* Informace o noteboocích s video portem naleznete v hardwarové části serveru TuxMobil, *http://tuxmobil.org/hardware.html.* Alternativně můžete namísto ZV portu použít USB port.

LCD displej

Tato kapitola není doposud hotova, měla by obsahovat informace o životnosti podsvícení, o roz-dílech mezi CRT a LCD displeji, o antialiasingu na LCD displejích, o normě ISO 13406-2 definují-cí povolenou chybovost pixelů a informace o běžných rozlišeních jako VGA, SVGA, XGA a dal-ších. Podívejte se také do kapitoly o spořičích obrazovky a do kapitoly o dotykových obrazov-kách ve třetím oddíle této příručky.

Displeje notebooků Aplikace

Program lcdtest, *http://www.brouhaha.com/~eric/software/lcdtest/,* slouží ke zobrazení různých testovacích obrazců na LCD displejích. S jeho pomocí snáze naleznete defektní pixely, ať už trva-le rozsvícené nebo trvale zhasnuté. Program používá knihovnu SDL a je testován pouze na Linu-xu, mohl by však fungovat i na jiných platformách.

Program DDCcontrol, *http://ddccontrol.sourceforge.net/,* slouží k softwarovému řízení parametrů monitoru, jako jsou jas nebo kontrast, není tedy nutné používat OSD (On Screen Display) a tla-čítka na monitoru.

Fonty

Font fat8x16-x-font, *http://www.guru-group.fi/~too/sw/fat8x16-x-font.readme,* je font s velikostí 8x16 pixelů s pevnou šířkou, který je vhodný k použití na fyzicky malých obrazovkách s velkým rozlišením. Jde například o 14" displeje s rozlišením 1400x1500 nebo 1600x1200. Zajímavý termi-nálový font pro X Window je například terminus (s podporou pro kódování iso-8859-2).

Displeje na PDA

Font pxl2000, *http://userpage.fu-berlin.de/~cantsin/homepage/computing/hacks/pxl2000/README.html,* je volně použitelný rastrový font s pevnou šířkou s kódováním ISO 8859-15 (tj. ISO 8859-1 se sym-bolem Eura) pro systém X Window. Je k dispozici v devíti velikostech: 4x8, 5x10, 6x12, 7x14, 8x16, 9x18, 10x20, 11x22 a 12x24 pixelů. Základními kritérii při jeho návrhu byly:

Čitelnost, je vhodný jako výchozí font, zejména na terminálech s inverzními barvami.
Optimalizace pro programátory s ohledem na dobrou rozlišitelnost znaků jako L, l, 1, 7, |, I, i a 0, O a dalších.
Úplná podpora znakové sady ISO 8859-15.
Mnoho různých velikostí k zajištění optimální konzistence mezi různými počítači s různými rozlišeními obrazovky (pokrývá cokoliv od PDA po 20" displeje).
Vhodnost ke zobrazení ASCII-artu, použití knihovny aview, sledování filmů v přehrávači mplayer s volbou -vo aa a podobně.
Čistý, minimalistický návrh, bez patiček, s oblými rohy. Jde o terminálový font, nesnaží se vypadat jako tiskový font se sníženým rozlišením.

Autor fontu Florian Cramer jej zahrnul do své konfigurace „anti-desktopu“ společně se správcem oken ratpoison a terminálem rxtv – podobné nastavení popisuje článek na serveru FreshMeat,
http://handhelds.freshmeat.net/articles/view/581/.

Zvuk

Ověření kompatibility s Linuxem

Jediná nám známá možnost ověření kompatibility daného zvukového zařízení s Linuxem je pře-ložit jádro s různými ovladači zvukových zařízení a vyzkoušet, zda některý ovladač zařízení nero-zezná. Nejlepší metoda je přeložit jednotlivé ovladače jako moduly, protože v případě modulů se dají snáze měnit parametry jako přerušení nebo vstupně-výstupní porty. Používáte-li jádra 2.2.x, přečtěte si dokument /usr/src/linux/Documentation/sound/Introduction Wade Hamptona, ten by vám měl pomoci zorientovat se v problematice zvukových zařízení pod Linuxem. Můžete také vyzkoušet některý z dále uvedených komerčních zvukových ovladačů. Kontrolu funkčnosti zvukového zařízení můžete provést například tak, že pomocí přehrávače jako xmms zkusíte pře-hrát některý ze zvuků v adresáři /usr/share/sounds.

Související dokumentace

Sound Howto, *http://www.tldp.org/HOWTO/Sound-HOWTO/index.html.*
Visual Bell miniHOWTO, *http://www.tldp.org/.*
Další užitečná zvuková HOWTO naleznete na stránkách Linux Audio Users Guide,

http://www.djcj.org/LAU/guide/.

Přehled zvukových ovladačů

ALSA, Advanced Linux Sound Architecture, *http://www.alsa-project.org/.* ALSA je plně modulární zvukový ovladač s podporou kerneld/kmod, nabízí kompatibilitu s většinou binárních OSS/Lite aplikací, obsahuje ALSA knihovnu (C, C++), která aplikacím zpřístup-ňuje jaderné rozhraní, a konečně obsahuje interaktivní konfigurační program ALSA Mana-ger. Od jádra 2.6 je ALSA součástí linuxového jádra.
OSS, UNIX Sound System Lite, *http://www.opensound.com/,* obsahuje komerční linuxové ovladače pro většinu rozšířených zvukových karet. Tyto ovladače podporují digitální audio, MIDI, syntezátory a mixéry na různých kartách. Ovladače vyhovují specifikaci OSS API a projekt nabízí příjemné grafické rozhraní, které usnadňuje instalaci a konfiguraci zvuko-vých karet. OSS podporuje více než 200 značkových zvukových karet, ovladače poskytují automatickou detekci, podporu PnP a podporu zvukových karet PCI.
Jako poslední pokus můžete vyzkoušet modul pcsnd, který emuluje zvukovou kartu pro-střednictvím integrovaného „reproduktoru“.

Další zvukové karty

VXPocket, *http://www.digigram.com/products/getinfo.htm?prod_key=9000,* vypadá jako kvalitní

zvuková karta pro notebooky s nekvalitní integrovanou zvukovou kartou. Další alternativou mohou být USB zařízení – nová jádra podporují většinu zvukových USB zařízení. Pří-kladem může být například Labtex Axis 712 Stereo Headset (sluchátka s mikrofonem), fungující v plně duplexním režimu. Další informace o různých zvukových USB zařízeních a jejich použitelnosti v Linu-xu naleznete na stránkách *http://www.qbik.ch/usb/devices/ a http://tuxmobil.org/usb_linux.html.*

Externí a interní CD mechaniky

O přehrávání CD a DVD disků v interních a externích CD/DVD mechanikách hovoříme dále v kapitole „Optické mechaniky (CD/DVD)“.

Klávesnice

Ověření kompatibility s Linuxem

Linux obvykle nemívá s klávesnicí problémy. Mohou však nastat dvě drobné komplikace. Ta první spočívá v tom, že nemusí fungovat program setleds. Druhá spočívá v tom, že mapování kláves nemu-sí odpovídat vašim potřebám. Někteří uživatelé Unixu a editoru vi jsou například zvyklí, že klávesu Ctrl mají nalevo od klávesy A. Na většině klávesnic je ale na tomto místě klávesa CapsLock. Pomo-cí příkazů xmodmap nebo loadkeys můžete klávesnici přemapovat. Některé notebooky (například Toshiba) umožňují klávesy CapsLock a Ctrl prohodit. Toto prohození popisuje Mark Alexander v konferenci linux-laptop – na RedHatu jde o jednořádkovou úpravu souboru /usr/lib/kbd/key-tables/us.map či jiného, na něž se odkazuje ze souboru /etc/sysconfig/keyboard:

```
*** us.map~ Tue Oct 31 14:00:07 1995 --- us.map Thu Aug 28 13:36:03 1997 *** 113,119 **** keycode 57 = space space
        control keycode 57 = nul
        alt keycode 57 = Meta_space ! keycode 58 = Caps_Lock keycode 59 = F1 F11 Console_13
        control keycode 59 = F1
        alt keycode 59 = Console_1 --- 113,119 ----keycode 57 = space space
        control keycode 57 = nul
        alt keycode 57 = Meta_space ! keycode 58 = Control keycode 59 = F1 F11 Console_13
        control keycode 59 = F1
        alt keycode 59 = Console_1
```

Externí klávesnice

Sekundární či externí klávesnici je možné připojit do portu PS/2 nebo do USB portu. Existuje dokonce notebook s oddělenou klávesnicí – Siemens Scenic Mobile 800. Počítač je s klávesnicí spojen infračerveným portem, netušíme však, zda toto řešení funguje v Linuxu.

Konfigurace externí USB klávesnice

Na architektuře PC nemusíte za určitých okolností vůbec potřebovat podporu USB klávesnice ope-račním systémem. Existuje několik BIOSů, které nativně podporují klávesnice připojené do koře-nového hubu na základní desce. Klávesnice ovšem nemusí fungovat v jiných hubech a nemusí se také dobře snášet s dalšími periferiemi, proto je rozumné nastavit podporu USB klávesnice v ope-račním systému. Obecně pokud máte Linux nastaven s podporou USB, musíte v něm nastavit rov-něž podporu USB klávesnice, protože USB podpora Linuxu vypne USB podporu BIOSu. Přímou podporu USB klávesnice v Linuxu budete potřebovat také v případě, že budete chtít využít různé „multimediální“ klávesy na některých typech USB klávesnic.
Při konfiguraci jádra musíte zapnout podporu USB Human Interface Device (HID) a podporu klá-vesnice. Nezapínejte podporu USB HIDBP klávesnice. Pokud příslušné komponenty přeložíte jakomoduly, musíte při startu systému zavést hid.o, input.o a keybdev.o.
Korektní detekci klávesnice jádrem poznáte ze záznamů v logu jádra.V tomto okamžiku byste měli být schopni používat USB klávesnici stejně jako obyčejnou. Zava-děč LILO nepodporuje USB, a pokud tedy nemáte BIOS s podporou USB klávesnice, nebudetemoci prostřednictvím USB klávesnice ovládat zavaděč.

Externí klávesnice PS/2

> Upozornění
> Nepřipojujte externí klávesnici v průběhu bootování a dejte si pozor na záměnu portů klá
> vesnice a myši. Na některých noteboocích tím můžete přivodit zatuhnutí notebooku.

Pro port PS/2 existuje tzv. Y-kabel, který umožňuje současné připojení externí myši i externí klá

vesnice, samozřejmě pokud to podporuje daný notebook.Existuje i adaptér z paralelního portu na AUX, *http://linuxconsole.sourceforge.net/input/adapters.html,* pro případ, že by vám jeden klávesnicový a jeden AUX port nestačily.

Pak může-te použít zmíněný adaptér a modul parkbd.

Doplňkové klávesy / horké klávesy

Související dokumentace

Keyboard and Console HOWTO, *http://tldp.org/HOWTO/Keyboard-and-Console-HOWTO.html*

Nástroje

Některé notebooky mají různé doplňkové klávesy – Internet, e-mail a podobně. Pokud pro ně linuxová jádra a XFree86/X.Org generují klávesové kódy, není problém použít programy hotke-ys nebo xmodmap (podrobnosti naleznete v manuálových stránkách těchto programů). Pokud ovšem Linux na tyto klávesy nijak nereaguje, musíte nejprve upravit jádro. I když netušíme, jak je to možné, některé nástroje dokážou na speciální klávesy reagovat i bez úpravy jádra. Můžete vyzkoušet program xhkeys, *http://www.geocities.com/wmalms/.* Tento nástroj umožňuje nastavit akce pro klávesy, jejichž význam není v X definován (například klávesa „menu“ na 105klávesové klávesnici, doplňkové klávesy na různých modelech klávesnic a noteboocích). Klávese či kombi-naci kláves (klávesa s předřaďovači) je možno přiřadit různé akce, což může být nějaká vestavě-ná operace, volání externí aplikace, vygenerování klávesové události (simulace stisku/uvolnění jiné klávesy) nebo vygenerování události myši (simulace stisku/uvolnění tlačítka myši).

> Tip
>
> Informace o neznámých událostech klávesnice nebo myši můžete zjistit v konzole pomo cí programů showkey a mev (ten pochází z balíku gpm). Tyto nástroje však nemusí dete-kovat všechny doplňkové klávesy.

Jednoduché nastavení speciálních funkčních kláves (jako jsou multimediální klávesy) umožňuje program keyTouch, *http://keytouch.sourceforge.net/.* Umožňuje nadefinovat, jaký program se má spustit při stisku té které klávesy. Pomocí editoru si uživatel může snadno vytvořit celou mapu

klávesnice pro svůj notebook.Démon akdaemon umožňuje používat některé doplňkové klávesy prostřednictvím zařízení, kterémusí být zpřístupněno buď dodávaným jaderným patchem (*http://sourceforge.net/projects/akdae-mon/*) nebo programem funkey (*http://rick.vanrein.org/linux/funkey/*).
Balík hotkeys (*http://ypwong.org/hotkeys/*) by měl podporovat různé multimediální klávesy.Speciální přístupové klávesy podporuje program LinEAK, *http://lineak.sourceforge.net/.* Taktovypadá příklad jeho konfiguračního souboru, lineakd.conf file:

```
# LinEAK Configuration file for Compaq Easy Access Key 2800 (6 keys)

# Global settings KeyboardType = CIKP800 CdromDevice = /dev/cdrom MixerDevice = /dev/mixer

# Specific keys of your keyboard internet = xosview search = kfind mail =
kmail multimedia = "artsdsp xmms" voldown = "aumix -v -2" volup =
"aumix -v +2"

# end lineakd.conf
```

Program xbindkeys, *http://freshmeat.net/projects/xbindkeys/,* asociuje v X Window klávesy nebo tlačítka myši s příkazy shellu. Po jednoduchém nastavení tak budete moci celou řadu příkazů spouštět přímo z klávesnice nebo tlačítky myši (například kombinací Ctrl+Alt+X můžete spustit xterm).

Program ACME, *http://devin.com/acme/,* je nástroj pro prostředí GNOME, který využívá různá mul-timediální tlačítka na noteboocích a internetových klávesnicích – hlasitost, jas, vypínání, vysunutí mechaniky, pošta, uspání, web, kalkulačka, přehrávání, pauza a spousta dalších. Funguje na všech platformách, kde funguje GNOME, tedy i na noteboocích. K ovládání hlasitosti využívá ovladače ALSA nebo OSS.

Pro některé série notebooků existují specializované linuxové nástroje, které zajišťují podporu speciálních kláves a některých dalších funkcí:

Pro některé modely Toshiba je určen nástroj toshutils Jonathana Buzzarda, *http://www. buzzard.me.uk/toshiba/index.html.*
Tclkeymon, *http://sourceforge.net/projects/tclkeymon/,* je démon pro notebooky Toshiba, který využívá služeb ACPI a rozšíření Toshiba ACPI. Sleduje funkční klávesy a specifické klávesy notebooků Toshiba (například tlačítka pro ovládání CD přehrávače) a správně na ně reaguje.
Tpcl, *http://tpctl.sourceforge.net/,* je konfigurační nástroj Thomase Hooda pro notebooky IBM ThinkPad (dnes již Lenovo).
Program ThinkPad Buttons, *http://savannah.nongnu.org/projects/tpb/,* zapíná speciální klá-vesy na klávesnici notebooků

ThinkPad. Jednotlivým klávesám je možné přiřadit různé programy. Program obsahuje také on-screen displej, na kterém zobrazuje nastavení hlasi-tosti, jasu, ztlumení zvuku a některé další údaje.

IBM ThinkPad Scroll Daemon, *http://rsim.cs.uiuc.edu/~sachs/tp-scroll/.*
Nástroj i8k, *http://people.debian.org/~dz/i8k/,* je určen pro notebooky DELL.
Ovladač hotkey, *http://www2.informatik.hu-berlin.de/~tauber/acerhk/,* je pro notebooky Acer.
OSL, *http://www.blinkenlights.ch/cgi-bin/fm.pl?get=osle,* je jednoduchý klient služby pbbut-tonsd (používané na noteboocích Apple pro přístup ke „speciálním klávesám", jako je nastavení hlasitosti, vysunutí mechaniky a podobně).
PBButtons, *http://pbbuttons.sourceforge.net/index.html,* zapíná horké klávesy na note-boocích Apple iBook/PowerBook/TiBook. Pokud víme, funguje dobře i na architektu-rách x86.
Ikeyd, *http://www.dreamind.de/linux/ikeyd,* je jednoduchý démon, který reaguje na kláve-sy ovládání hlasitosti a vysunutí mechaniky na iBooku/TiBooku.
Jogdiald, *http://perso.orange.fr/pascal.brisset/vaio/,* zajišťuje podporu speciálních kláves na noteboocích Sony Jog-Dial.
Omke, *http://sourceforge.net/projects/omke/,* je sada malých programů a patchů pro nasta-vení některých funkcí notebooků HP OmniBook (typicky funkcí, které HP nedokumentu-je), například zapnutí a vypnutí speciálních multimediálních kláves. Tento nástroj funguje i na některých noteboocích Toshiba.

Funkční klávesa

Funkční klávesa (obvykle označená jako Fn) typicky slouží k zapnutí simulované numerické klá-vesnice, která je na stolních počítačích realizována samostatným blokem kláves. Pokud numeric-kou klávesnici potřebujete a její simulace vám nevyhovuje, prodávají se samostatné numerické bloky pro porty PS/2 i USB. V kombinaci s různými klávesami F*x* slouží tato klávesa k nastavo-vání jasu, hlasitosti, zamčení klávesnice, přepínání mezi interním a externím displejem a aktivaci uspání. Některé kombinace fungují v Linuxu bez jakéhokoliv nastavování, chcete-li využívat i jiné, použijte stejné nástroje, které jsme popsali v předchozí kapitole.

Vypínač

Vypínací tlačítko často mívá i jiné funkce, kromě zapnutí a vypnutí slouží například k probuzení notebooku z uspaného režimu. Notebook probudíte krátkým stlačením tlačítka, pokud je podrží-te stlačené déle (více než 5 sekund), dojde k vypnutí notebooku.

Na moderních noteboocích s podporou ACPI je možné vypnutí provést i prostřednictvím rozhra-ní /proc/apci/.

Doplňkové LED

Některé notebooky obsahují různé doplňkové LED, například LED pro signalizaci příchozí pošty. Nalezli jsme dva programy, které by s těmito LED měly pracovat – getmail a fujled, oba z pro-jektu The Led Project. Užitečný může být také nástroj setleds, součást projektu Linux Console Tools, *http://lct.sourceforge.net/.*

Numerická klávesnice

Na klávesnicích stolních počítačů je numerická klávesnice obvykle tvořena samostatným blokem kláves, na noteboocích však tento samostatný blok kláves není. Jeho funkce bývá emulována růz-nými způsoby, například pomocí klávesy Fn nebo NumLock. Existují i samostatné numerické klá-vesové bloky připojitelné přes PS/2 (nebo USB či RS232).

Jak už bylo řečeno dříve, numerickou klávesnici potřebujete, pokud chcete změnit rozlišení X Window klávesami Ctrl+Alt+Num + a Ctrl+Alt+Num-. Pokud tuto kombinaci na notebooku ne-jste schopni vytvořit nebo je to příliš pracné, můžete použít program gvidm, *http://packages. debian.org/gvidm.* Program gvidm vám nabídne seznam dostupných grafických režimů a umožní uživateli jeden vybrat. Díky tomu je ideální kandidát na spouštění z nabídky nebo prostřednictvím doplňkové klávesy a nemusíte tak mít trvale spuštěn nějaký applet pro změny rozlišení. Pokud máte připojeno dva nebo více monitorů, program vám umožní si mezi nimi vybírat. Můžete také použít příkazy xvidtune [-next | -prev]. Chcete-li zjistit aktuální nastavené rozlišení, můžete pou-žít příkazy xvidtune nebo xwininfo -root.

Ukazovací zařízení – myši a další

Ověření kompatibility s Linuxem

Myš můžete ověřit příkazem mev z balíku GPM.

Související dokumentace

3-button Mouse HOWTO, *http://tldp.org/HOWTO/3-Button-Mouse.html,* týkající se sério-vých myší.
Kernel HOWTO, *http://www.digitalhermit.com/linux/Kernel-Build-HOWTO.html.*

Alternativní ukazovací zařízení

Trackpad nebo touchpad, používané ve většině dnešních notebooků.
Trackball, například u modelů Compaq LTE.
Pop-up-Mouse, například u modelu HP OmniBook 800.
Trackpoint nebo Mouse-Pin, například u notebooků IBM ThinkPad a Toshiba.
Třítlačítkové myši v modelech IBM Thinkpad 600 a vyšších a v některých modelech Com-paq, například Armada M700.
Dotyková obrazovka, například v noteboocích Fujitsu-Siemens, v tabletových PC a PDA.

Myši PS/2

Většina myší v noteboocích je typu PS/2. S těmito myšmi můžete komunikovat prostřednictvím zařízení /dev/psaux nebo /dev/psmouse. Pokud používáte X Window, musíte mít v souboru /etc/X11/XF86Config nastavena odpovídající zařízení a odpovídající protokol. Ve starších ver-zích vznikaly problémy při současném běhu X Window a GPM. V nových verzích však už k tomu-to problému nedochází.

V souvislosti s funkcí Emulate3Buttons je praktičtější nastavit souběh 100 ms namísto hodnoty 50 ms, která je výchozím nastavením. V XFree86 3.x stačí upravit soubor /etc/X11/XF86Config:

```
Section "Pointer"
        ...
        Emulate3Buttons
        Emulate3Timeout 100
        ...

EndSection
```

V XFree86 4.x upravujeme soubor /etc/X11/XF86Config-4:

```
Section "InputDevice" ... Option "Emulate3Timeout" "100" Option "Emulate3Buttons" "true" ...
EndSection
```

Stejně bude vypadat také konfigurace X.org v souboru /etc/X11/xorg.conf.

Touchpad

Touchpad většinou funguje se zařízením /dev/psaux a protokolem PS/2 (můžete také vyzkoušet protokol GlidePointPS/2).Ovladač Synaptics TouchPad, *http://web.telia.com/~u89404340/touchpad/index.html,* nabízí celouřadu funkcí (některé z nich musí být podporovány přímo touchpadem, například dotek víceprsty):

Pohyb s nastavitelnou nelineární akcelerací a rychlostí (volby MinSpeed, MaxSpeed, AccelFactor).
Události myši generované krátkým dotekem touchpadu (volby MaxTapTime a MaxTapMove).
Události dvojího klepnutí generované dvojím dotekem touchpadu.
Přesouvání prostřednictvím krátkého stlačení a následného posuvu po touchpadu.
Události prostředního a pravého tlačítka generované klepnutím v horním a dolním rohu touchpadu (volba Edges).
Rolování (události čtvrtého a pátého tlačítka myši) přesouváním prstu v pravé části touchpadu (volby Edges, VertScrollDelta).
Tlačítka nahoru a dolů generují události čtvrtého a pátého tlačítka myši.
Nastavitelná detekce prstu (volba Finger).
Dotyk více prsty – dva prsty znamenají prostřední tlačítko, tři prsty pravé tlačítko.
Online konfigurace přes sdílenou paměť (ve vývoji, volba SHMConfig).

Společně s ovladačem je distribuován i příkaz synclient (v SuSE Linuxu přinejmenším do verze 9.3 však tento příkaz chybí). Tento příkaz umožňuje za běhu zjišťovat a modifikovat nastavení Touchpadu.

> Tip
>
> Dotek jedním, dvěma nebo třemi prsty současně generuje událost stisku levého, prostředního, respektive pravého tlačítka myši.

Existuje ještě další ovladač, Synaptics Touchpad Linux Driver, *http://www.compass.com/synaptics/.* S ním související program tpconfig podporuje ukazovací zařízení používaná v noteboocích Acer, Compaq, Dell, Gateway, Olivetti, Texas Instruments, Winbook a dalších.

Společnosti Dell a Sony začaly používat touchpad a touchstick od ALPS. Jsou jimi vybaveny při-nejmenším modely Dell Latitude CPx a Sony VAIO. Správce programu tpconfig Bruce Kall píše: „Tpconfig je v současné době nepodporuje, snažím se ale od ALPS získat příslušné API a pod-poru zahrnu v příští verzi. Dell používá také ALPS GlideStick ve středu klávesnice (podobně, jako to mají některé modely IBM Thinkpad). Hodlám implementovat i podporu vypnutí reakce na dotyk GlideSticku. Nevím, jak

vás, ale mě přivádí k zuřivosti, když se náhodným dotykem ozna-čují věci ve chvíli, kdy to nechci.“

Tpconfig je řádkový nástroj k nastavení parametrů touchpadu Synaptics a glidepadu/stickpoin-teru ALPS. Většina lidí je primárně používá k vypnutí „dotekového režimu“ touchpadů na note-boocích.

Tpconfig se používá jako příkazový nástroj. Port PS/2 nepodporuje sdílení, takže tpconfig nelze použít, pokud je nahrán jiný ovladač myši (například gpm). Znamená to také, že tpconfig nelze použít, pokud jsou spuštěna X Window. Doporučené použití je tedy spuštění z inicializačního skriptu dříve, než se spustí gpm.

Ne všechny touchpady pocházejí od Synapticsu, například některé modely Gateway používají EZ-Pad a jistě existují i jiné typy. Nástrojem TPREV.EXE (*http://www.synaptics.com/decaf/utilities/ tprev.exe*) ověříte, že skutečně máte touchpad od Synapticsu.

Novější verze balíku gpm (verze >= 1.8) obsahují i ovladač pro touchpady Synaptics. Tento ovla-dač vyvinul H. Davies (*hdavies@ameritech.net*). Nemusíte používat režim kompatibility s PS/2, místo něj můžete použít nativní režim touchpadu, který nabízí i další funkce.

Kromě převodu posunu prstu na pohyb myši a podporu tlačítek touchpadu nabízí nová verze i další funkce (citujeme z dokumentu README):

Klepnutí na touchpad generuje klepnutí levým tlačítkem myši.
Klepnutí následované pohybem generuje akci pohybu myši se stlačeným tlačítkem.
Klepnutí do jednoho z rohů je ve výchozí konfiguraci obsluhováno tak, že pravý horní roh vyvolává klepnutí prostředním tlačítkem, pravý spodní roh klepnutí pravým tlačítkem.
Větší tlak na touchpad zrychluje pohyb kurzoru.
Pokud se prstem dostanete na okraj touchpadu, pohyb myši bude pokračovat započatým směrem, takže nemusíte prst zvedat a znovu s ním pohybovat. Rychlost tohoto automatického pohybu závisí na tlaku.

Tyto funkce je možné zapnout a vypnout a u řady z nich je možné nastavovat parametry rychlosti
a času podle potřeb uživatele. Ovladač gpm je nejznámější jako nástroj pro textový režim. To je samozřejmě pravda, lze jej ale použít jako vstupní zařízení i v X Window; gpm se používá jako opakovač, takže můžete použí-vat vestavěný touchpad se všemi jeho funkcemi a zároveň i třítlačítkovou sériovou myš. Všechno funguje hladce, X Window načítají události myši z pojmenované fronty /dev/gpmdata pomocí protokolu, kterému rozumí (v mém případě to je 5bajtový *Mouse-Systems-Compatible*). Většina tří-tlačítkových myší používá tento výchozí protokol. Jediné, co je zapotřebí, je tedy jednoduchá rekonfigurace v souboru XF86Config (xorg.conf) a samozřejmě správné spuštění gpm.

Ovladače gpm byste měli spouštět následujícím způsobem: /usr/bin/gpm -t synps2 -M -t ms -m /dev/ttyS0. Touchpad i myš budou fungovat jak v textovém, tak v grafickém režimu. Pojme-novanou rouru /dev/gpmdata musíte vytvořit sami.
Současný dotek dvěma prsty simuluje stisk prostředního tlačítka myši i na některých touchpadech
Logitech.
Autorem většiny textu této kapitoly je Geert van der Plas.

Jog-Dial

Jog-Dial je vstupní zařízení používané v noteboocích Sony VAIO. Ovladač napsal Takaya Kinjo, *http://nerv-un.net/~dragorn/jogmouse/.* V souboru spicdriver/Makefile budete nejspíš muset změnit dvě věci:

rozšířit příznak CCFLAG o -D_LOOSE_KERNEL_NAMES
a rozšířit příznak CCFLAG o -I/usr/src/linux-<verze-jádra>/include.

Soubor README je v japonštině, postup překladu vypadá takto:

$ tar xvzf jogutils.tar.gz $ cd jogutils $ make $ su # mknod /dev/spic c 60 0 # modprobe spicdriver/spicdriver # exit $ cp jogapp/rcfile ~/.jogapprc $ jogapp/jogapp

Ishikawa Mutsumi napsal ovladač *http://perso.orange.fr/pascal.brisset/vaio/,* který běží v uživatel-ském prostoru a nepotřebuje tak žádný jaderný modul. Existuje i neoficiální balíček pro Debian:

*http://hanzubon.org/Linux/Debian/jogdiald.*Nástroj rsjog, *http://linuxbrit.co.uk/rsjog/,* je rozšířením nástroje sjog, *http://sjog.sourceforge.net/.*

Dotykové obrazovky

Pokud víme, jediné modernější notebooky, které používají dotekovou obrazovku, jsou FujitsuBiblo 112/142 (též MC 30) a Palmax PD 1000/1100 (též IPC 1000/1000).Na stránkách Joe Pfeiffera, *http://www.cs.nmsu.edu/~pfeiffer/,* naleznete poslední verzi ovladače.Další přehled ovladačů naleznete také na stránce *http://tuxmobil.org/touch_laptops.html.*

Pera a ukazovátka

IBM a Toshiba začínají místo mousepadů či trackballů používat ve svých noteboocích ukazovací pera.

> Tip
>
> Na toto vstupní zařízení si musíte zvyknout. Pomáhá opírat dlaň o přední část notebooku, vhodné je také snížit rychlost myši.

Externí myš

Větší míru pohodlí získáte při použití externí třítlačítkové myši. Typicky ji připojíte do sériového portu, PS/2 nebo USB podle toho, jakou máte myš a jaké porty váš notebook nabízí. Většinou s tím není žádný problém. Na starších systémech budete muset po připojení myši restartovat X server, novější si již umí (slespoň u USB myší) tuto událost obsloužit bez restartu a začnou myš pou-žívat téměř okamžitě. V případě připojení PS/2 a sériové myši za chodu doporučujeme restart X serveru vždy.

Myš PS/2

Pro porty PS/2 existuje Y-kabel, který umožňuje současné připojení externí myši i externí klávesnice, pokud notebook takové uspořádání podporuje.

> Upozornění
>
> Nepřipojujte externí myš při zapnutém notebooku, nespleťte si port myši a port klávesni
> ce. Může to vést až k nutnosti tvrdého restartu notebooku.

Myš s kolečkem

Program imwheel, *http://imwheel.sourceforge.net/,* umožňuje používat kolečko myši v X Window k posouvání obsahu okna a ke generování klávesových událostí. Běží na pozadí jako démon a vyžaduje jen minimální zásahy do konfigurace X Window. Program podporuje i čtyř- a více-tla-čítkové myši a Alps Glidepad Taps. Program imwheel obsahuje modifikovanou verzi gpm. Dá se říci, že dnes již není jeho použití nutné a kolečko je nativně podporováno ovladači a knihovnami systému X.org.

Dokument Wheel Mouse FAQ, *http://colas.nahaboo.net/mouse-wheel-scroll/,* popisuje, jak nastavit obsluhu kolečka v prostředí X Window. Dalším vhodným dokumentem je XFree86 4.x – Mouse Docs, *http://www.xfree86.org/current/mouse.html.*

USB myš

Následující text je převzat z dokumentu The Linux USB Sub-System Brada Hardse. Než se pustíte do ručního nastavování, vyzkoušejte, zda je vůbec nutné – většina nových distribucí umí USB myši detekovat a nastavit za chodu X Window „sama“ pomocí nástrojů, jako je hotplug.

Konfigurace USB HID (Human Interface Device)

Obecná konfigurace HID

Existují dva způsoby, jak zajistit podporu USB myši a klávesnice – samostatný bootovací protokol, anebo plná podpora ovladačem HID. Metoda bootovacího protokolu je obecně horší, proto si popíšeme plnohodnotné řešení. Bootovací protokol je v zásadě vhodný pouze pro embedded systémy a jiné systémy s omezenými prostředky, u nichž není plná funkčnost klávesnice a myši potřebná.

Je důležité vědět, že ovladač HID obsluhuje ta zařízení (nebo přesněji ta rozhraní na jednotlivých zařízeních), která odpovídají specifikaci HID. Tato specifikace ovšem neříká nic o tom, co má ovladač s informacemi přijatými od HID zařízení udělat nebo odkud se berou informace posílané na zařízení – to totiž zjevně závisí na operačním systému a na tom, co má zařízení dělat. Linux na úrovni jádra podporuje čtyři rozhraní k HID zařízením: klávesnici, myš, joystick a obecné rozhraní, takzvané rozhraní událostí.

Konfigurace HID myši

Při konfiguraci jádra musíte zapnout podporu USB Human Interface Device (HID) a podporu myši. Nezapínejte podporu USB HIDBP myši. Jádro přeložte a nainstalujte běžným postupem. Pokud podporu instalujete jako moduly, musíte zavést moduly input.o, hid.o a mousedev.o.
Zapojte USB myš a ověřte si, že ji jádro nalezlo. Pokud se v logu jádra neobjeví zpráva, podívej
te se na změny v /proc/bus/usb/devices.USB podporuje více identických zařízení, takže můžete mít připojeno více myší. Každá myš můžebýt obsluhována samostatně, ale je možné vstupy od všech myší smíchat do jednoho. Ve většiněpřípadů budeme chtít vstupy míchat. Nejprve musíme vytvořit uzel zařízení pro míchané myši.Obvykle se uzel vytváří v adresáři /dev/input.

Použijte následující příkazy:

mkdir /dev/input mknod /dev/input/mice c 13 63

Tip Pokud si nejste jisti, zda konfigurujete správné zařízení, použijte příkaz cat /dev/input/mice (nebo jiný název zařízení). Jestliže jste zvolili správné zařízení, budou se při pohybu myší a mačkání tlačítek vypisovat různé podivné znaky.

Budete-li chtít myš používat v prostředí X Window, máte několik možností. Volba závisí na pou-žívané verzi XFree86/X.org a na tom, zda USB myš (či myši) používáte jako jediné zařízení, anebo zda chcete použít USB myš společně s jiným ukazovacím zařízením.

Musíte upravit soubor XF86Config (typicky /usr/X11R6/lib/X11/XF86Config nebo /etc/X11/XF86Config). Pokud používáte XFree86 verze 4.0 a vyšší, přidejte sekci InputDevice, která bude vypadat taktonějak:

```
Section "InputDevice"
    Identifier      "USB Mice"
      Driver        "mouse"
      Option        "Protocol"      "IMPS/2"
      Option        "Device"        "/dev/input/mice"
  EndSection
```

Jestliže máte myš s kolečkem, použijte nastavení:

```
Section "InputDevice"
Identifier "USB Mice"
Driver "mouse"
Option "Protocol" "IMPS/2"
Option "Device" "/dev/input/mice"
Option "ZAxisMapping" "4 5"
        Option "Buttons" "5" EndSection
```

Podrobnější vysvětlení a další příklady naleznete v dokumentaci k XFree86/X.org, *http://www.xfree86.org/current/mouse.html, http://wiki.x.org/wiki/Documentation.* Dále musíte přidat záznam do každé odpovídající sekce ServerLayout. Ty se za normálních okolností nacházejí na konci konfiguračního souboru. Pokud máte pouze USB myš (či myši), nahraďte řádek s textem „CorePointer“ tímto řádkem:

```
InputDevice "USB Mice" "CorePointer"
```

Pokud chcete používat USB myš (či myši) zároveň s jiným vstupním zařízením, přidejte (nenahrazujte) následující řádek:

```
InputDevice "USB Mice" "SendCoreEvents"
```

Jestliže používáte pouze USB myš (či myši) v prostředí XFree86 3.3, upravte sekci Pointer takto:

```
Section "Pointer"
    Protocol        "IMPS/2"
     Device         "/dev/input/mice"
  EndSection
```

Pokud chcete používat USB myš (či myši) a zároveň jiné ukazovací zařízení v XFree86 3.3, musí-te použít rozšíření XInput. Ponechejte existující sekci Pointer (pokud provádíte počáteční nasta-vení, tak ji vhodným způsobem změňte podle potřeb druhého ukazovacího zařízení) a přidejte následující záznam (kdekoliv to dává smysl, nejlépe do části Input devices):

```
Section "Xinput"
  SubSection "Mouse" DeviceName "USB Mice" Protocol "IMPS/2" Port "/dev/input/mice" AlwaysCore
    EndSubSection EndSection
```

Restartujte X server. Pokud vám myš nebude fungovat, stiskem Ctrl+Alt+F1 se přepnete na virtu-ální terminál, odkud můžete X server ukončit a prozkoumat jeho chybová hlášení. Pokud chcete myš používat pod gpm, spusťte (nebo zastavte a restartuje, pokud už běží) gpm s následujícími volbami: gpm -m /dev/input/mice -t imps2 (samozřejmě jako superuživatel). Můžete také upravit inicializační soubory tak, aby se tyto volby používaly vždy. Typicky půjde o některý ze rc souborů v adresáři /etc/rc.d/. Pokud používáte současně USB myš (nebo myši) a nějaké jiné ukazovací zařízení, můžete gpm použít v režimu opakovače. Máte-li připojenu PS/2 myš jako /dev/psaux a USB myš (myši) jako /dev/input/mice, pak by měl posloužit následující příkaz: gpm -m /dev/input/mice -t imps2 -M -m /dev/psaux -t ps2 -R imps2. Tímto příkazem budete výstup předávat do FIFO /dev/gpm-data, kterou nemusíte vytvářet předem. Tu pak zadáte jako „zařízení“ myši programům mimo X Window a obě myši budou fungovat současně.

Volba Popis

Sériová myš MicroSoft kompatibilní. PS/2 nebo C&T 82C710. Sběrnicová myš Logitech. Sběrnicová myš ATI XL. Sběrnicová myš MicroSoft. Sériová myš Mouse Systems. Staré myši. Protokol Mouse Man, sériová myš Logitech. Třítlačítková myš SUN. Intellimouse s kolečkem, sériový port. Intellimouse s kolečkem, PS/2. PnP myši, alternativa k Série MM. Nejstarší dvoutlačítkové myši.

Parametry voleb a programu

Twiddler

Gpm obsahuje ovladač pro Twiddler na sériovém portu. Informace o tomto zařízení naleznete na stránkách Handykey Corporation, *http://www.handykey.com/*.

Macintosh PowerBook

PowerBook má trackpad s jediným tlačítkem, můžete ale připojit externí USB myš s více tlačítky. Obvykle se vybrané klávesy z klávesnice namapují na prostřední a pravé tlačítko myši. Návod k tomuto postupu by měl být součástí vaší linuxové distribuce (nejde o problematiku týkající se specificky notebooků, protože všechny myši Apple jsou jednotlačítkové).

Pokud používáte server Xpmac, výchozí nastavení je option-1 a option-2, můžete je změnit para-metry -middlekey <kód_klávesy> -rightkey <kód_klávesy> nebo volbou -nooptionmouse pokud nechcete klávesu Option používat.

Pokud používáte XFree86, předáváte jádru parametry adb_buttons=<prostřední_tlačítko>, <pravé_tlačítko>. Já používám parametr adb_buttons=58,55, čímž na tlačítka myši mapuji klá-vesy Option a Apple/command. Kódy zvolených kláves můžete zjistit například programem xev.

Advanced Power Management – APM Ověření kompatibility s Linuxem

Začněte přečtením dokumentu Battery Powered miniHOWTO, *http://tldp.org/HOWTO/html_sing-le/Battery-Powered/*.

Aby APM fungovalo, musí firmware počítače implementovat specifikaci APM, Linux podporuje verze 1.0 až 1.2 tohoto standardu. Pro komunikaci s Linuxem musí APM BIOS podporovat 32bito-vé připojení v chráněném režimu.

Informace o APM BOISu na svém počítači můžete zjistit příkazem dmesg | grep apm avsou-boru /proc/apm. APM se používá spíše ve starších počítačích, nové stroje používají APCI, viz následující kapitolu.

Úvod

Podpora APM se skládá ze dvou částí: podpora *v jádře* a podpora *v uživatelském prostoru*.

Podpora v jádře

Abyste mohli použít příslušné konfigurační volby jádra, potřebujete jádro s přeloženým ovlada-čem APM. Většina distribucí nepoužívá jádra se zapnutým ovladačem APM, takže jej budete muset buď zapnout při bootování nebo si přeložit vlastní jádro. Podrobnosti naleznete v dokumentu Ker-nel HOWTO, *http://www.digitalhermit.com/linux/Kernel-Build-HOWTO.html,* nebo v návodu k vaší distribuci.

Ovladač APM může být přeložen jako modul, ale toto řešení se nedoporučuje, protože celá řada ovladačů vypíná své APM funkce v případě, že při jejich zavádění není ještě zaveden ovladač APM.

Dostupné volby APM (podrobnosti viz Documentation/Configure.help ve zdrojovém stromu jádra):

CONFIG_APM_IGNORE_USER_SUSPEND – obezlička pro některé notebooky NEC Versa M.
CONFIG_APM_DO_ENABLE – zapíná funkce APM při bootování.
CONFIG_APM_CPU_IDLE – pokud jádro nemá co na práci, procesor přejde do režimu snížené spotřeby.
CONFIG_APM_DISPLAY_BLANK – některé notebooky využívají této volby k vypnutí pod-svícení displeje v okamžiku, kdy se vypne obrazovka virtuální konzoly. Toto nastavení fun-guje pouze na virtuálních konzolách, k vypnutí podsvícení nedojde, pokud používáte X Window.
CONFIG_APM_POWER_OFF – po zadání příkazu halt vypne počítač. Na většině notebo-oků tato volba bez problémů funguje.
CONFIG_APM_IGNORE_MULTIPLE_SUSPEND – obezlička pro IBM ThinkPad 560.
CONFIG_APM_IGNORE_SUSPEND_BOUNCE – obezlička pro Dell Inspiron 3200 a někte-ré další.
CONFIG_APM_RTC_IS_GMT – ukládá čas ve formátu GMT. Obecně se doporučuje mít čas v BIOSu uložen v GMT.
CONFIG_APM_ALLOW_INTS – řeší problémy s uspáním na disk u některých notebooků, například u většiny novějších IBM ThinkPad.
CONFIG_SMP – podpora SMP pro systémy s více procesory. Pokud máte systém s jediným procesorem (což je většina osobních počítačů), volbu nezapínejte. Výchozí nastavení je nicméně zapnuté. Máme informace, že podpora SMP může s APM kolidovat.

Podle dokumentace jádra nabízí ovladač APM následující funkce: Po operaci USER RESUME dojde k reinicializaci systémového času, zařízení /proc/apm poskytuje informace o stavu baterií, uživa-telské programy mohou přijímat APM události (například změna stavu napájení).

Podpora v uživatelském prostoru

Nejdůležitějším nástrojem uživatelského prostoru je apmd, démon, který obsluhuje události APM. Pokud používáte jádro 2.2 a vyšší a máte chuť experimentovat, bude vás zajímat, že Gabor Kuti (*seasons@falcon.sch.bme.hu*) vytvořil patch, který umožňuje hibernovat linuxový systém na disk
i v případě, že APM BIOS tuto funkci přímo nepodporuje. Jestliže notebook přímo podporuje uspání na disk, nebudete tento patch potřebovat. Další informace naleznete v dokumentu Battery Powered miniHOWTO na adrese
http://tldp.org/HOWTO/html_single/Battery-Powered/.

Démon apmd nabízí jiné následující věci:

apmd(8): Do syslogu zaznamenává stav baterií a spouští proxy skript, který provádí potřeb-né operace před uspáním a po probuzení.
amp(1): Vypíše aktuální stav baterií nebo uspí počítač.
apmsleep(1): Uspí počítač na omezenou dobu.
xapm(1x): Měřič baterií pro X Window.
libapm.a: Knihovna pro tvorbu APM aplikací.

Některé APM firmwary neobnovují při probuzení správně stav zvukového mixéru. Řešením je pou-žití skriptu, který po probuzení nastaví mixér správně (např. pomocí nástrojů, jako je aumix nebo alsamixer).

Z manuálové stránky apmsleep(1): Některé počítače, zejména notebooky, se dokážou probudit v nastavený čas. Apmsleep lze použít k nastavení budíku v RTC a přepnutí do uspaného režimu. Přerušení od RTC způsobí probuzení počítače. Program tuto událost rozezná podle skokové změny jaderného času a správně se ukončí. Pokud do minuty nedojde ke skoku v čase nebo se něco nezdaří, návratový kód bude nenulový. Apmsleep je součást balíku apmd.V roce 2001 napsal Richard Gooch jednoduchou alternativu k démonu apmd, kterou naleznetev balíku pmutils, *http://www.atnf.csiro.au/people/rgooch/linux/*.

Možná vás bude zajímat i nástroj apmcd (crontab řízení prostřednictvím apm), ftp://ftp. binary9.net/pub/linux/. Tento nástroj napsal Nicolas J. Leon, *nicholas@binary9.net.*

Úskalí

Pokud na stejném počítači používáte více operačních systémů, zajistěte, aby jejich funkce uspání či hibernace nezapisovaly na diskový oddíl, který používá Linux.

Řešení problémů

Pokud vám notebook funguje s jádry 2.0 a nikoliv s řadou 2.2, zkuste tuto radu Klause Frankena, *kfr@klaus.franken.de*: „Výchozí nastavení se v řadě 2.2 změnilo. V inicializačních skriptech hle-dejte halt a změňte to na halt -p nebo poweroff. Viz man halt, pokud tuto volbu nemáte k dis-pozici, potřebujete novější verzi.“ Tu najdete v balíku SysVinit.

Na některých novějších noteboocích (například HP Omnibook 4150, 366 MHz) může při přístupu k /proc/apm dojít k jaderné chybě general protection fault: f000. Vysvětluje ji Stephen Roth-well, *http://www.canb.auug.org.au/~sfr/*: „Je to problém APM BIOSu, který se v chráněném reži-mu pokouší přistupovat k segmentu reálného režimu, jde tedy o chybu BIOSu. Už jsme se s ní několikrát setkali, ale ve většině případů to bylo ve vypínací sekvenci BIOSu, kde jsme schopni chybě předejít tím, že před samotným vypnutím přejdeme do reálného režimu. Za normálního běhu to ale udělat nejde.“

Další rady uvádí jaderná dokumentace Documentation/Configure.help: „Několik dalších věcí, které byste měli vyzkoušet, pokud registrujete zjevně náhodné podivné problémy:

Zkontrolujte, že máte dostatek odkládacího prostoru a že je zapnutý (swapon -s).
Předejte jádru volby no-hlt.
Zapněte v jádru emulaci koprocesoru a předejte mu volbu no387.
Předejte jádru volbu floppy=nodma.
Předejte jádru volbu mem=4M (čímž vypnete celou paměť kromě prvních 4 MB).
Nepoužívejte přetaktovaný procesor (zejména na mobilních zařízeních to není vhodné).
Přečtěte si dokument *http://www.bitwizard.nl/sig11/*.
V nastaveních BIOSu vypněte cache. Neplatí pro cache procesoru!
Nainstalujte větrák na grafickou kartu nebo vyměňte video RAM (nevhodné pro mobilní zařízení).
Nainstalujte lepší větrák na procesor (nevhodné pro mobilní zařízení).
Vyměňte RAM (nevhodné pro mobilní zařízení).

Vyměňte základní desku (nevhodné pro mobilní zařízení).“

APM a PCMCIA

Z dokumentu PCMCIA HOWTO, *http://pcmcia-cs.sourceforge.net/ftp/doc/PCMCIA-HOWTO.html*: „Pokud máte jádro nastaveno s podporou APM, je možné přeložit s podporou APM i služby PCMCIA. ... Pokud bude v systému detekována kompatibilní verze, PCMCIA moduly si automa-ticky nastaví APM. Bez ohledu na to, zda máte APM nastaveno, můžete před uspáním notebooku použít příkaz cardctl suspend a po probuzení příkaz cardctl resume – tím dojde ke korektní-mu vypnutí a zapnutí PCMCIA karet. Tyto příkazy nebudou fungovat u aktivovaného modemu, protože sériový ovladač není schopen uložit a obnovit provozní nastavení modemu. Na některých systémech se APM chová nestabilně. Pokud narazíte na problémy mezi APM a PCMCIA, snažte se problém zúžit na konkrétní problémový ovladač. Některé ovladače, zejména SCSI PCMCIA, se nedokážou obnovit z uspaného stavu. Pokud tedy používáte SCSI PCMCIA kartu, před uspáním vždy použijte příkaz cardctl eject.“

APM a probuzení X Window

Některé počítače mají APM firmware, který nedokáže při uspání správně uložit a obnovit registry grafického čipu. Starší verze X serveru XFree86 neobnoví po probuzení správně stav obrazovky, tento problém je popsán v dokumentu *http://tuxmobil.org/index.html.* Novější verze Xfree86/X.org už většinou fungují správně.

V některých případech dobře nespolupracuje X server a APM, může to vést dokonce až k zatuh-nutí počítače. Doporučení Steva Radera: „Na některých systémech může při zadání příkazu apm -s X server vytuhnout. V takovém případě si můžete pomoci tak, že se nejprve přepnete na vir-tuální terminál a pak zadáte chvt 1; apm -s. Tyto příkazy mám uloženy ve skriptu my-suspend a pak zadávám xapmload --click-command my-suspend.“

Softwarové uspání

Projekt Software Suspend, *http://sourceforge.net/projects/swsusp,* umožňuje uspání počítače i bez podpory APM. Počítač můžete uspat buď stiskem Sysrq-D nebo příkazy swsusp nebo shutdown -z (je nutný patch do sysvinit). Vytvoří se obraz, který se uloží do aktivního swapovacího sou-boru. Při příštím bootování jádro tento obraz naleznete, obnoví stav paměti a pokračuje se v práci tam, kde jste počítač uspali. Pokud byste při bootování nechtěli obnovit předchozí stav, použijte volbu noresume.

Softwarové uspání může být lepší než hibernace (uspání do paměti), protože tak můžete uspat Linux, nabootovat Microsoft Windows, několika příkazy je shodit a pak pokračovat v práci v Linu-xu tam, kde jste přestali. Hibernací takového chování nedosáhnete, protože počítač se vždy pro-budí tam, kde byl uspán, ať už to bylo ve Windows nebo v Linuxu. Softwarové uspání je tedy vhodné řešení, pokud chcete uložit rozpracovaný stav v Linuxu a udělat něco ve Windows.
V jádrech 2.6 je softwarové uspání už přímo součástí jádra. Naleznete je v části Power Manage-ment. Existuje i backport pro jádra 2.4. Původní kód softwarového uspání vytvořili Gabor Kuti a Pavel Machek už v roce 1998, pro jádra 2.6 tak existují už tři různé implementace, všechny vycházející z tohoto původního kódu. Jejich

srovnání naleznete v dokumentu *http://www.suspend2.net/features.*Projekt Software Suspend 2 obsahuje dlouhý seznam funkcí, včetně možnosti stiskem klávesyEscape přerušit uspávání, komprese ukládaného obrazu šetřící čas i prostor, pružné architekturypodporující doplňky, podpory počítačů s velkou pamětí, preempcí a SMP.

Tipy a triky Stav baterií v textové konzole

Následujícími úpravami souboru .bashrc můžete zapnout zobrazování stavu baterií v promptu příkazového řádku.
Používáte-li APM

```
export PS1="\$(cat /proc/apm | awk '{print \$7}') \h:\w\$ "
```

Používáte-li ACPI

Color the bash prompt in function of the percentage of battery # with acpi subsystem. # Based on the originally apm based script that has been posted # on debian-laptop by # Jason Kraftcheck <kraftche at cae.wisc.edu>. # # This script is licensed under the GNU GPL version 2 or later, # see /usr/share/common-licences/GPL on a Debian system or # http://www.gnu.org/copyleft/gpl.html on the web.

(c) 2003 Fabio 'farnis' Sirna <farnis at libero dot it>

```
function acpi_percent() { if
[ `cat /proc/acpi/battery/BAT0/state | grep present: |cut -d\
-f18` = "yes" ]; then
{ CAPACITY=`cat /proc/acpi/battery/BAT0/info |grep
"design capacity:"|cut -d\
                -f11` LEVEL=`cat /proc/acpi/battery/BAT0/state | grep remaining|cut -d\ -f8` ACPI_PERCENT=`echo $(( $LEVEL * 100 /
```

```
      $CAPACITY ))` if [ "$LEVEL" = "$CAPACITY" ]; then
        echo FULL
       else
       echo $ACPI_PERCENT%
       fi

    }
  else echo "NO BATTERY"
  fi

}

function acpi_charge()
{ ACPI_CHARGE=`cat /proc/acpi/ac_adapter/AC/state | cut -d\ -f20` case $ACPI_CHARGE in
                *on-line*)
                    ACPI_CHARGE="+" ;;
                    *off-line*)

                  ACPI_CHARGE="-" ;;
          esac
          echo $ACPI_CHARGE

}

            function acpi_color() { if [ "$(acpi_charge)" = "+" ]; then { if [ `cat
                 /proc/acpi/battery/BAT0/state | grep present: |cut -d\  -f18` =
                 "no" ]; then
               echo "0;31"else echo "1;32"
             fi
           }
           else

              case $(acpi_percent) in
                   10?%) echo "0;32" ;;9?%) echo "0;32" ;;8?%) echo "0;32" ;;7?%) echo "0;32" ;;6?%) echo "0;32" ;;5?%) echo "0;32" ;;4?%) echo
                     "0;33" ;;3?%) echo "0;33" ;;2?%) echo "0;33" ;;1?%) echo "0;31" ;;
                       ?%) echo "0;31;5" ;;*) echo "0;35" ;;

              esac
         fi
         }


           function acpi_color_prompt { PS1='\[\e[$(acpi_color)m\][$
                (acpi_charge)$(acpi_percent)][\t] \u:\w\$>\[\e[0;37m\] ' }

         # linux consoleif [ "$TERM" = "linux" ];
         thenPROMPT_COMMAND=acpi_color_promptfi

         function echo_acpi
         {
         echo -n "($(acpi_charge)$(acpi_percent)) "
         }
```

Debian GNU/Linux

Všechna „normální“ jádra Debian GNU/Linuxu podporují APM, v konfiguračním souboru zavadě-če (například /etc/lilo.conf) je pouze nutné přidat následující řádek:

```
append="apm=on"
```

Pokud máte v jádře přeloženu podporu ACPI i APM, můžete vhodnými změnami následujícího řádku v zavaděči experimentovat s obojím. Nelze totiž současně používat ACPI i APM, podrob-nosti naleznete v dokumentaci jádra.

```
append="acpi=off apm=on"
```

Totéž bude platit i pro ostatní distribuce používající LILO.

ACPI

Související dokumentace

ACPI HOWTO 1, *http://tldp.org/HOWTO/ACPI-HOWTO/.*
ACPI HOWTO 2, *http://www.columbia.edu/~ariel/acpi/acpi_howto.txt.* Tento dokument popisuje překlad, instalaci a použití ovladače ACPI a souvisejících aplikací.
ACPI HOWTO 3, *http://www.cpqlinux.com/acpi-howto.html.*
Projekt ACPI4Linux, *http://acpi.sourceforge.net/wiki/.*
ACPI Info, *http://www.acpi.info/.*
Kapitola „CPU“ této příručky.

Podrobnosti

ACPI znamená *Advanced Configuration and Power Interface*. Jde o společnou specifikaci společ-ností Toshiba, Intel a Microsoft. Mimo jiné definuje i správu napájení, proto bývá často srovnává-no s APM.

Pokud máte jádro přeloženo s podporou APM i s podporou ACPI, můžete v konfiguračním sou-boru zavaděče (například /etc/lilo.conf) uvést následující volbu. Současné použití APM a ACPI není možné.

```
append="acpi=on apm=off"
```

Projekt Linux ACPI, *http://sourceforge.net/projects/acpi*, vyvíjí základní komponenty ACPI pro Linux. Patří mezi ně obecný parser tabulky ACPI, interpretr AML, ovladače sběrnice a zařízení, politiky, uživatelské rozhraní a pomocné nástroje.

Epplet E-AcpiPower, *http://www.netego.de/hpc?p=acpipower&l=en*, je založen na E-Power. Čte stav baterií pomocí nového jaderného modulu acpi, takže je mnohem přesnější a spolehlivější než star-ší metoda založená na APM.

Na adrese *http://rffr.de/acpi naleznete TCL/TK skript*, který umožňuje prostřednictvím ACPI v gra-fickém rozhraní nastavit výkon procesoru.Skript acpi.py, *http://www.iapp.de/~riemer/projects/acpi.py*, představuje uniformní a platformněnezávislé rozhraní k ACPI.

Linux ACPI Client, *http://grahame.angrygoats.net/acpi.shtml*, je řádkový nástroj podobný příkazu apm, který umožňuje zjistit stav baterií, napájení a chlazení.

Power Management Unit – PMU (PowerBook)

PowerBooky nepodporují specifikaci APM, pro správu napájení (PMU) používají vlastní protokol. Existuje GPL démon pmud, který umí se správou napájení komunikovat, dokáže zjistit stav bate-rií, uspat počítač a nastavovat různé úrovně spotřeby. Napsal jej Stephan Leemburg. Existuje také starší nástroj snooze, sloužící k uspání PowerBooku. Funkce těchto programů jsou nyní zahrnu-ty v projektu PBButtons, *http://pbbuttons.sourceforge.net/index.html.*

Pokud notebook nevypnete úplně a pouze jej uspíte, po probuzení se provedou všechny úlohy naplánované v cronu na dobu uspání.

Baterie

Informace o různých typech baterií naleznete v předchozí kapitole, věnované základním hardwa

rovým informacím.Další informace se nacházejí v dokumentu Battery Powered miniHOWTO, *http://tldp.org/HOWTO/-Battery-Powered/*, a na serveru TuxMobil, *http://tuxmobil.org/mobile_battery.html.* Na tomto ser-veru můžete získat i informace o alternativních způsobech napájení mobilních zařízení,*http://tuxmobil.org/energy_laptops.html.* Dalším zdrojem informací je také dokument Battery FAQ,*http://www.technick.net/public/code/cp_dpage.php?aiocp_dp=guide_bpw2_00_faq.*
Stephen Rothwell navrhl patch, *http://www.canb.auug.org.au/~sfr/*, umožňující jadernému APM podporovat více baterií.Citujeme ze stránky mobile-update: „Vybijte baterii. Pokud notebook na baterii běží jen asi 20minut, pravděpodobně trpí paměťovým efektem. Většina notebooků nevybíjí baterie dokonale.Úplného vybití lze dosáhnout pomocí nějakého zařízení s malou spotřebou, například počítačo-vým ventilátorem. Tím se paměťové efekty odstraňují. Tento postup je vhodný i pro baterie LiIon,přestože paměťovým efektem příliš netrpí (návod k IBM Thinkpadu doporučuje každých několikměsíců třikrát zopakovat cyklus úplného vybití a nabití).“

> Upozornění
>
> Manipulace s bateriemi může být riskantní. Ověřte si, zda souhlasí napětí baterie a připo jovaného ventilátoru. Mně tento postup nicméně funguje.

Balíček battery-stats, *http://karl.jorgensen.com/battery-stats/*, zjišťuje informace o nabití baterií v notebooku. Obsahuje také jednoduchý nástroj pro grafické zobrazení nabití baterií v čase a dete-kuje anomální chování, které může být příznakem končící životnosti baterií. Program neví nic
o elektrochemických procesech uvnitř baterie, proto se také nepokouší o žádnou předpověď. Pokud ale znáte vlastnosti používaných baterií, můžete ze zjištěných údajů odvodit, zda jsou v pořádku. Program používá APM, doposud nepodporuje ACPI.

IBAM (Intelligent Battery Monitor), *http://ibam.sourceforge.net/*, je nástroj pro sledování baterií, který používá statistické a adaptivní metody k přesnému odhadnutí zbývající životnosti baterie a potřebné doby nabíjení. I tento program používá pouze APM, nepodporuje ACPI.

Booker C. Bense upravil program *rclock*, *http://www-leland.stanford.edu/~bbense/toys/*, takže na ciferníku zobrazuje i stav baterií. Prostředí KDE i Gnome má vlastní programy pro zobrazení stavu baterií (např. KLaptop).

Podpora smart baterií

Balík sbsutils, *https://sourceforge.net/projects/sbs-linux/*, je sada programů, které zajišťují podporu technologie Smart Battery, používané v některých noteboocích.

Pamě

Některé notebooky se bohužel vyrábějí s proprietárními paměťovými čipy. Paměti tak nejsou zaměnitelné mezi různými modely. Tento trend se ale postupně mění. V některých noteboocích je instalace paměti značně komplikovaná, protože je nutné rozebrat celou skříň. I toto se ale postupně mění. U moderních notebooků jsou paměti přístupné po odstranění krytu vespod skříně, případně stačí jen odmontovat klávesnici.

Zařízení Plug-and-Play (PnP)

Projekt Plug-and-Play for Linux (podrobnosti viz *http://www.linux.org/*) zajišťuje konzistentní pod-poru PnP a semi-PnP zařízení v Linuxu. Umožňuje ovladačům hardwarových komponentnakonfigurovat příslušný hardware podle pokynů ovladače PnP. Ovladač zná konfiguracejednotlivých zařízení, reaguje na jejich připojení a odpojení a dokáže tak na tyto situacesprávně reagovat. Dalším užitečným balíkem jsou *ISA PnP Tools*.Balík ovladače PCMCIA od verze 3.1.0 obsahuje nástroje lspnp a setpnp, umožňující manipulo-vat s nastavením PnP.

Dokovací stanice, replikátory portů

Definice

Nejprve několik definicí. Mezi *dokovací stanicí* a *replikátorem portů* jsou následující rozdíly: *Dokovací stanicí* rozumíme zařízení, které obsahuje sloty pro připojení přídavných karet, pevné-ho disku a podobně, a rozšiřuje tak možnosti připojeného notebooku. *Replikátor portů* pouze kopíruje porty, jimiž je notebook vybaven.

Další řešení

Osobně dokovací stanici nepoužívám. Jsou poměrně drahé a nevidím žádné zásadní výhody.Manipulace s kabely je bez ní trochu obtížnější, ale za ušetřené peníze to stojí. Rozumné využitímůže mít například v kancelářském prostředí, kde máte trvalé připojení k síti nebo kde potřebu-jete rozšiřující sloty k připojení například nějakých exotických SCSI zařízení.

Všechny dokovací stanice jsou proprietární, takže jakmile změníte notebook, musíte změniti dokovací stanici. Jedinou výjimkou je dokovací stanice IRDocking IR-660 společnosti Tekram,*http://www.tekram.com/*, která se k notebooku připojuje IrDA portem. Stanice nabízí konektorypro připojení 10Base-T (RJ-45), PS/2 klávesnice, PS/2 myši, 25pinový paralelní port tiskárnya napájení. VGA port neobsahuje. Stanice by měla spolupracovat s linuxovou podporou IrDA.
Alternativně doporučuji koupit stolní počítač a propojit jej s notebookem přes síť.Další možností je externí monitor, klávesnice a myš. Pokud notebook obsahuje PS/2, můžete sikoupit levný Y-kabel, který umožňuje připojit myš i klávesnici současně.

Způsoby připojení dokovací stanice

Pokud je nám známo, existují čtyři různé metody připojení dokovací stanice:
SCSI port (zřídkavé řešení).
Paralelní port.
Proprietární dokovací port (nejobvyklejší řešení).
USB (často používané nezávislými výrobci).

Martin J. Evans říká: „Hlavní problém při použití dokovací stanice spočívá v tom, aby operační systém správně rozeznal, že je notebook připojen do doku. Můžete to poznat například pomocí detekce zařízení v /proc. Pak vám stačí několik skriptů, které notebook při připojení a odpojení správně nastaví.
Podporu pro hardware dokovací stanice můžete mít přeloženu v podobě modulů, a nikoliv přímo v jádře. Tím ušetříte paměť, vše ale závisí na tom, jak často notebook k dokovací stanici připojujete.

Podpora dalších disků připojených ke SCSI kartě dokovací stanice:

Přeložte podporu SCSI karty jako modul nebo ji přeložte přímo v jádře.
Vytvořte body připojení v /etc/fstab, použijte ale příznak noauto, aby nedocházelo k jejich automatickému připojování při zapnutí.
Po zapojení notebooku do dokovací sta-nice explicitně připojte příslušné oddíly.

Podpora dalších síťových adaptérů v dokovací stanici:Můžete použít podobný způsob, jako jsme popisovali v souvislosti s grafickými kartami. V rc skrip-
tech ověřte obsah souborového systému /proc, podle toho poznáte, zda je notebook připojen

k doku, a pak můžete správně nastavit síťové připojení.“
Jakmile máte potřebné informace, můžete použít skript podobný následujícími příkladu a nakon
figurovat zařízení v dokovací stanici:

```
# check, if Laptop is in docking-station (4 PCMCIA slots available) # or if it is standalone (2 slots available) # Start after cardmgr has started # # Friedhelm Kueck mailto:fk_AT_impress.de # 08-Sep-1998 # # Find No. of Sockets SOCKETS=`tail -1 /var/run/stab | cut -d ":" -f 1` case "$SOCKETS" in "Socket 3") echo Laptop is in Dockingstation ... echo Disabeling internal LCD Display for X11 echo cp /etc/XF86Config_extern /etc/XF86Config # # Setup of PCMCIA Network Interface after start of cardmgr # echo echo "Setting up eth0 for use at Network ..." echo /sbin/ifconfig eth0 10.1.9.5 netmask 255.255.0.0 broadcast 10.1.255.255 /sbin/route add -net 10.1.0.0 gw 10.1.9.5 /sbin/route add default gw 10.1.10.1 ;;

"Socket 1") echo Laptop is standalone echo Disabling external Monitor for X11 cp /etc/X11/XF86Config_intern /etc/X11/XF86Config echo echo Network device NOT setup ;; esac
```

Replikátory USB portů

Funkčnost jsme testovali pomocí dokovací stanice Typhoon USB 2.0 7in1 společnosti Anubis, *http://www.anubisline.com/*. Toto zařízení by se mělo ve skutečnosti označovat jako replikátor portů, protože neobsahuje žádné rozšiřující sloty. Nemá ani VGA konektor pro připojení externí-ho displeje, tím je vybaveno jen velmi málo USB dokovacích stanic. Bylo by zajímavé zjistit, zda VGA na těchto stanicích funguje. Testovali jsme s notebookem Compaq M700 (USB 1.1) a vlast-ním jádrem 2.6.1. Replikátor nefunguje s Apple PowerBookem G4.

Jednotlivé porty replikátoru se v Linuxu chovají takto:

USB 2.0 – funguje s externím diskem i myší bez dalšího nastavování.
Klávesnice PS/2 – funguje.
Myš PS/2 – funguje, v jádrech 2.6 je ale nutné správně nastavit protokol pomocí paramet-ru psmouse_proto=imps.
Sériový port – testováno se sériovou myší, nefunguje, zařízení /dev/ttyUSB0 bylo přiřa-zeno.
Paralelní port – testováno, přiřadilo se zařízení /dev/usb/usblp0, tiskárna HP LaserJet 2100, funguje.
Síťový port – zavede se usbnet, vytvoří se zařízení eth1, ifconfig funguje.
Přenosový port – funguje s modulem usbnet, příkazem ifconfig usb0 nastavíte síťové roz-hraní, netestováno.

Výpis dmesg při připojení stanice vypadá takto:

```
hub 1-0:1.0: new USB device on port 1, assigned address 26 hub 1-1:1.0: USB hub found hub 1-1:1.0: 4 ports detected hub 1-1:1.0: new USB device on port 3, assigned address 27 hub 1-1.3:1.0: USB hub found hub 1-1.3:1.0: 4 ports detected hub 1-1:1.0: new USB device on port 4, assigned address 28 eth1: register usbnet at usb-0000:00:07.2-1.4, ASIX AX8817x USB 2.0 Ethernet hub 1-1.3:1.0: new USB device on port 1, assigned address 29 usb0: register usbnet at usb-0000:00:07.2-1.3.1, Prolific PL-2301/PL-2302 hub 1-1.3:1.0: new USB device on port 2, assigned address 30 drivers/usb/class/usblp.c: usblp0: USB Bidirectional printer dev 30 if 0 alt 1
  proto 2 vid 0x067B pid 0x2305 hub 1-1.3:1.0: new USB device on port 3, assigned address 31 pl2303 1-1.3.3:1.0: PL-2303 converter detected usb 1-1.3.3: PL-2303 converter now attached to ttyUSB0 (or usb/tts/0 for devfs) hub 1-1.3:1.0: new USB device on port 4, assigned address 32 HID device not claimed by input or hiddev hid: probe of 1-1.3.4:1.0 failed with error -5 input: Composite USB PS2 Converter USB to PS2 Adaptor v1.09 on usb-0000:00:07.2-1.3.4 HID device not claimed by input or hiddev hid: probe of 1-1.3.4:1.1 failed with error -5 input: Composite USB PS2 Converter USB to PS2 Adaptor v1.09 on usb-0000:00:07.2-1.3.4
```

Síová připojení

Související dokumentace

PLIP miniHOWTO, *http://tldp.org/HOWTO/PLIP.html.*
Networking HOWTO, *http://tldp.org/HOWTO/NET3-4-HOWTO.html.*

Ethernet HOWTO, *http://tldp.org/HOWTO/Ethernet-HOWTO.html.*

Způsoby připojení

Většina moderních notebooků má vestavěnou síťovou kartu. V této kapitole popisujeme některé způsoby, jak k síti připojit starší notebooky bez vestavěné síťové karty.

Síové karty PCMCIA

Pokud váš notebook podporuje PCMCIA, je to nejjednodušší a nejrychlejší způsob, jak se připojit k síti. Před koupí síťové karty si ověřte, že ji Linux podporuje.

Sériový nullmodemový kabel

Jde asi o nejlevnější způsob, jak propojit dva počítače, je však také nejpomalejší. Spojení můžete navázat pomocí protokolů PPP nebo SLIP.

Síový adaptér do paralelního portu

Na stránkách *http://www.home.unix-ag.org/nils/accton_linux.html* naleznete popis ethernetového adaptéru připojovaného do paralelního portu, který dosahuje přenosovou rychlost zhruba 110 kB/s.

Paralelní „nullmodemový“ kabel

Je rychlejší než propojení přes sériový port. Čipsety některých notebooků neumožňují použít pro-tokol PLIP. Podrobnosti naleznete v dokumentu PLIP HOWTO, *http://tldp.org/HOWTO/PLIP.html.*

Síové karty v dokovací stanici

S tímto řešením nemáme žádné praktické zkušenosti.

Wake-on-LAN

Technologie Wake-on-LAN funguje u některých notebooků s vestavěnou síťovou kartou. Jde o obecné označení technologie „magického paketu“, vyvinuté původně společností AMD. Funkč-ně se podobá signálu „wake on ring“ na PCMCIA modemech. Základní myšlenka spočívá v tom, že síťový adaptér je trvale v režimu velmi nízkého příkonu a čeká na speciální paket, po jehož při-jetí notebook probudí. Pomocí nástroje etherwake nebo perlového skriptu Wakeonlan můžete vygenerovat příslušný magický paket a probudit tak vzdálený počítač.

Vestavěný modem

Typy modemů

Existují tři základní typy modemů – interní, v podobě PCMCIA karty nebo externí modem připo-jovaný přes sériový port. Některé interní modemy, takzvané WinModemy, v Linuxu občas nefun-gují, což je dáno jejich nestandardním hardwaru. V takovém případě musíte použít buď modem na PCMCIA kartě nebo externí modem, sériový nebo USB. O použití některých interních mode-mů hovoří dokument LinModem HOWTO, *http://walbran.org/sean/linux/linmodem-howto.html*, Seana Walbrana. Další informace o podpoře modemů v různých noteboocích naleznete v doku-mentech *http://tuxmobil.org/modem_linux.html* a *http://tuxmobil.org/minipci_linux.html.*

Citujeme z Kernel FAQ: „Proč nejsou podporovány WinModemy? Problémem je neexistující spe-cifikace k těmto zařízením. Většina společností, které takzvané WinModemy vyrábí, odmítá poskytnout specifikaci, díky níž by je bylo možné použít i v jiných operačních systémech, než jsou Windows. Základní problém spočívá v tom, že tato zařízení nefungují jako klasické modemy – nemají DSP čip, veškerou práci vykonává CPU. Nemůžete s nimi proto komunikovat jako s běž-nými modemy a potřebujete speciální ovladač běžící jako realtime úloha, jinak bude docházet ke ztrátám dat.

Tyto *Win*modemy jsou prostě lobotimizovanou verzí klasických modemů, očekávají, že přemýšlet za ně budou Windows. Nemáte-li Windows, nebudou fungovat.“ Situace se poslední dobou naštěstí výrazně zlepšila a linuxové ovladače jsou dostupné pro hodně winmodemů, resp. jejich čipsetů.

Na serveru TuxMobil naleznete stránku s informacemi o noteboocích s interními modemy, *http://tuxmobil.org/hardware.html.* Teoreticky vzato by mělo být možné tyto modemy provozovat s emulátory jako wine nebo WMware, nemáme však o tom žádné zprávy.

Na stránkách *http://linmodems.org/* se dozvíte, zda je určitý modem „opravdový“ nebo ne, a nalez-nete zde také návody na zprovoznění těch několika málo winmodemů, pro něž existují linuxové ovladače.

Na stránkách SuSE Labs naleznete alfa verzi ovladače pro winmodemy Lucent, k dispozici je také diagnostický nástroj LTModem, *http://www.close.u-net.com/.* Pro interní modemy PCI existuje binární ovladač LucentPCI, naleznete jej na adrese

http://linmodems.org/.

Úskalí

Upozornění: Dejte si pozor na různé typy telefonních linek, analogové a ISDN. Analogový modem nelze připojit k ISDN lince a naopak. Připojením k nevhodné lince můžete v nejhorším případě modem zničit.

Upozornění: Pokud máte počítač vybaven vestavěným modemem i vestavěnou síťovou kartou, dávejte si pozor, ať kabely zapojíte do správné zástrčky, jinak můžete poškodit hardware. Pakety na PPP spojení můžete sledovat programem pppstats. Grafické znázornění přenosu na PPP lince ukazuje program pload, založený na widgetech athena, s minimálními nároky na pro-cesor. Naleznete jej na adrese *http://www.engr.utk.edu/~mdsmith/pload/.* Podobnou funkci nabízí i nástroj KPPP.

GPRS

GPRS je zkratka pro General Packet Radio Service, službu provozovanou v sítích GSM a TDMA na celém světě. Umožňuje připojení k Internetu prostřednictvím GSM (nebo TDMA) telefonu. Díky tomu můžete notebook s minimálními náklady připojit k Internetu. Esa Turtianen, *etu@dna.fi*, a Jari Arkko, j*ari@arkko.com*, vytvořili dokument GPRS HOWTO, *http://turtiainen.dna.fi/GPRS-HOWTO.*

SCSI

Ověření kompatibility s Linuxem

Pokud si nejste jisti volbou SCSI podpory, přeložte jádro se všemi SCSI ovladači jako moduly. Pak můžete zkoušet jeden po druhém, až naleznete ten správný.

Související dokumentace

SCSI 2.4 HOWTO, *http://tldp.org/HOWTO/SCSI-2.4-HOWTO/index.html.*

Přehled

Žádný ze stávajících x86 notebooků nemá SCSI disk. Existují dva modely s vestavěným SCSI por-tem: Texas Instruments TI4000 a HP OmniBook 800. SCSI disk měly staré modely Apple Power-book Duo.

Pokud potřebujete podporu SCSI u jiných modelů, můžete ji zajistit pomocí SCSI PCMCIA karty nebo SCSI adaptérem v dokovací stanici.

Universal Serial Bus – USB

Ověření kompatibility s Linuxem

Informace o USB řadiči ve svém počítači získáte příkazem cat /proc/pci, informace o připoje-ných USB zařízeních příkazem cat /proc/bus/usb/devices.

Různé

Všechny novější notebooky jsou vybaveny USB porty. K USB sběrnici lze připojit různá zařízení, ne všechna jsou ale v Linuxu plně podporována. Lze připojit například klávesnici, myš, tiskárnu, tablet, fotoaparát, webovou kameru, přehrávač MP3, modem, bezdrátovou síťovou kartu, audio zařízení, skener, disková zařízení, disketovou mechaniku, ZIP a CD-ROM mechaniky, BlueTooth i Ethernet adaptéry, sériový port, joystick a další.

Řadu informací naleznete na stránkách USB Linux, *http://www.linux-usb.org/.* Na serveru TuxMo-bil rovněž shromažďujeme informace o noteboocích s USB rozhraním, *http://tuxmobil.org/ hardware.html.*

> Upozornění
>
> Napájení některých zařízení přes USB rozhraní notebooku nemusí postačovat, například u některých webových kamer nebo pevných disků. Záleží vždy na konkrétním notebooku a konkrétním zařízení. U stolních počítačů nebývá s napájením USB zařízení problém, u notebooků to ale problém může být.

FireWire – IEEE1394

Firewire, též IEEE1394 nebo iLink, je vysokorychlostní sériová sběrnice vyvinutá původně společ-ností Apple Computer. V současnosti se nejčastěji používá u digitálního videa, má však i mnoho jiných použití. Stejně jako USB je Firewire sériový protokol s podporou hot-swapu. Firewire pod-poruje mnohem vyšší rychlosti než USB 1 (USB 2 je v rychlosti srovnatelné). V Linuxu zajišťuje podporu IEEE1394 zařízení subsystém IEEE1394, *http://www.linux1394.org/.* Skládá se z jaderné části a uživatelských aplikací.Informace o noteboocích s IEEE1394 naleznete na serveru TuxMobil, *http://tuxmobil.org/hardware.html.*

Disketová mechanika

Ověření kompatibility s Linuxem

S připojením disketové mechaniky k linuxovému notebooku nebývá většinou problém. Ne vždy vám ale budou fungovat všechny funkce. Zjistili jsme například, že v notebooku HP OmniBook 800 pomocí příkazu superformat (z balíčku fdutils) nenaformátujeme disketu na více než 1,44 MB. Problémy mohou nastat také v případech, kdy se vylučuje současné připojení disketové mechaniky a CD mechaniky nebo pokud máte PCMCIA disketovou mechaniku (například u note-booku Toshiba Libretto 100). U starších notebooků může být problém, pokud podporují pouze mechaniky s kapacitou 720 kB. Všechny distribuce jsou dostupné pouze na disketách s kapacitou 1,44 MB. I v takovém případě by ale nemělo být těžké Linux nainstalovat některým z postupů popsaných v kapitole o instalaci. Některé disketové mechaniky, například u notebooků IBM ThinkPad, vyžadují nabootovat jádro se speciálními parametry. Jejich popis naleznete v doku-mentaci k jádru nebo na manuálové stránce man bootparam.

Optické mechaniky (CD/DVD)

CD-ROM Související dokumentace

CDROM HOWTO, *http://tldp.org/HOWTO/CDROM-HOWTO/.*
CD Writing HOWTO, *http://tldp.org/HOWTO/CD-Writing-HOWTO.html.*

Úvod

Většina dnešních notebooků obsahuje CD mechaniku. Pokud jsou disketová mechanika a CD mechanika zapojovány do společné šachty, nelze je používat současně. Řada výrobců (například HP nebo Dell) však vyrábí kabely, které umožňují připojení disketové mechaniky k paralelnímu portu. V některých případech je CD mechanika realizována jako externí PCMCIA zařízení (napří-klad Sony), SCSI zařízení (například HP OmniBook 800), USB zařízení (Sony) nebo Firewire (Sony VAIO VX71P). U takovýchto externích zařízení může být problematické z nich Linux nainstalovat.

Některé diskmany Sony jsou vybaveny portem pro připojení k počítači nebo přímo SCSI portem. V interních CD mechanikách lze bez potíží přehrávat i zvuková CD. Pozor ale:

> Tip
>
> Některé notebooky mají externí CD mechaniku. Pokud v ní chcete přehrávat zvuková CD, musíte si pořídit speciální kabel pro propojení zvukového výstupu mechaniky se zvukovým vstupem notebooku. Další možností je použití digitálního přehrávání (data jsou přenášena po sběrnici), touto možností disponuje například přehrávač Kscd.

CD-RW

Většina notebooků bývá vybavena interní nebo externí CD vypalovačkou. Interní typicky funguje bez potíží, viz dokument CD Writing HOWTO, *http://tldp.org/HOWTO/CD-Writing-HOWTO.html*, s externími vypalovačkami (PCMCIA, FireWire, USB) mohou být různé potíže a jejich zprovozně-ní může být obtížnější.

DVD mechaniky

Program regionset, *http://linvdr.org/projects/regionset/*, umožňuje zobrazit a změnit nastavení regionu DVD mechaniky.K přehrávání DVD můžete použít například programy VideoLAN, *http://www.videolan.org/*, neboMPlayer, *http://www.mplayerhq.hu.*

Ovladač UDF, *http://www.bitwizard.nl/udf/*, je nový standard souborového systému pro CDROM,používaný na DVD discích. Je zamýšlen také jako náhrada dnes používaného souborového systémuISO9660 na CD. Multimediální DVD pracují právě se souborovým systémem UDF. K přehrávání DVDpotřebujete DVD mechaniku, jaderný ovladač mechaniky, přehrávač MPEG videa a ovladač soubo-rového systému UDF. Některá DVD mohou obsahovat souborový systém ISO9660.

Formáty DVD:

DVD-5 4,4 GB, cca 2 hodiny videa. DVD-9 8,5 GB, cca 4 hodiny videa.

DVD-10 9,4 GB, cca 4,5 hodiny videa. DVD-18 17 GB, cca 8 hodin videa.

Pevné disky

Ověření kompatibility s Linuxem

Užitečné programy jsou hdparm, dmesg, fsck a fdisk.

Nástroje

Balík smartmontools, *http://smartmontools.sourceforge.net/*, obsahuje dva nástroje (smartctl a smartd), které slouží k řízení a sledování diskových systémů pomocí technologie Self-Monito-ring, Analysis and Reporting Technology System (SMART), která je podporována většinou moder-ních ATA a SCSI disků. Ve většině případů tyto nástroje předem upozorní na degradaci a selhání disku.

Nástroj hddtemp, *http://www.guzu.net/linux/hddtemp.php*, čte teplotu disků s podporou techno-logie S.M.A.R.T.

Mechanické provedení

Pevné disky do notebooků se pokud víme vyrábějí výhradně ve formátu 2,5", jejich výška ale může být různá:

18 mm: Používané ve většině notebooků vyrobených před rokem 1996.
12,7 mm: Dozvěděli jsme se o této velikosti bez dalších podrobností.
11 mm: Používané po roce 1996.
9 mm: Většina notebooků včetně subnotebooků dnes používá 9 mm vysoké disky.
9,5 mm: Toshiba Libretto L70 a L100.
8,45 mm: Toshiba Libretto 20, 30, 50 a 60.
6,35 mm: Toshiba Libretto L1000.

Za určitých okolností lze použít i disk s trochu jinými rozměry, než je originál.Některé notebooky mají vyjímatelný disk v šuplíku, například KAPOK 9600D. Pokud víme, žádnýnotebook není vybaven SCSI disky.

Nástroje výrobců

Někteří výrobci poskytují vlastní nástroje ke změně parametrů disku. Například Hitachi nabízí Drive Fitness Test, *http://www.hitachigst.com/hdd/support/download.htm*, který představuje rych-lou a spolehlivou metodu testování SCSI a IDE disků, včetně disků SATA. Při analýze disku se data čtou bez přepsání zaznamenaných údajů. (Některé z funkcí tohoto programu však data na disku přepíší.) Na stejné adrese naleznete i Feature Tool, což je v DOSu bootovatelný nástroj, který mění nastavení různých parametrů ATA.

Hot-swap zařízení

Některé notebooky (zejména ty dražší) mívají volnou pozici, do níž je možné připojit sekundární disk nebo CD/DVD mechaniku. Každý výrobce tuto vlastnost označuje jinak, například MultiBay nebo SelectBay. Manipulaci s těmito za běhu výměnnými zařízeními zvládají různé linuxové nástroje.

Program thotswap je součástí balíku Toshiba Linux Utilities, *http://www.buzzard.me.uk/toshiba/index.html*, a umožňuje za běhu měnit zařízení v šachtě SelectBay.Nástroj hotswap umožňuje registrovat a deregistrovat hot-swap IDE zařízení. Je určen pro note-booky s hardwarovou podporou výměny disků za běhu. Nástroj je potřebný pouze při manipula-ci s IDE disky, při vkládání disketových mechanik a baterií není zapotřebí.

Nástroj hdparm pro správu pevných disků rovněž podporuje hot-swap zařízení.Do některých šachet je možné místo disku vložit druhou jednotku baterií.

Bezdrátové sítě – WLAN

Související dokumentace

Wireless HOWTO 1, *http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Linux. Wireless.drivers.html.*
Wireless HOWTO 2, *http://www.linux-wlan.org/.*
Wireless HOWTO 3, *http://www.fuw.edu.pl/~pliszka/hints/wireless.html.*

Úvod

Řada notebooků je dnes vybavena bezdrátovým zařízením podporujícím protokolovou rodinu 802.11. Tato zařízení jsou založena buď na miniPCI nebo na PCMCIA. Můžete to ověřit pomocí příkazů lspci nebo cardctl ident. Externí WLAN adaptéry se vyrábějí v provedení PCMCIA, CF karet nebo jako USB zařízení.

Podrobnosti v další verzi tohoto dokumentu.

BlueTooth

Některé notebooky jsou vybaveny vestavěnou podporou bezdrátového protokolu BlueTooth. S touto technologií nemáme prozatím moc zkušeností, ale podle ohlasů se zdá, že funguje celkem bezproblémově.

Infračervený port

Ověření kompatibility s Linuxem

Chcete-li v Linuxu používat IrDA port svého notebooku, můžete použít buď režim StandardInfra-Red (SIR) nebo FastIntraRed (FIR).

SIR

Až do rychlosti 115 200 b/s emuluje infračervený port chování sériového portu s řadičem 16550A UART. Toto zařízení detekuje sériový ovladač jádra při bootování nebo ve chvíli, kdy zavedete modul serial. Máte-li infračervený port v BIOSu zapnutý, na většině notebooků se obje-ví přibližně následující hlášení jádra:

Serial driver version 4.25 with no serial options enabled ttyS00 at 0x03f8 (irq = 4) is a 16550A #first serial port /dev/ttyS0 ttyS01 at 0x3000 (irq = 10) is a 16550A #e.g. infrared port ttyS02 at 0x0300 (irq = 3) is a 16550A #e.g. PCMCIA modem port

FIR

Pokud chcete používat rychlosti až do 4 Mb/s, musíte mít počítač s čipem FIR. Pro tento čip potře-bujete mít v jádře specifický ovladač, a potřebujete tedy přesně zjistit, jakým čipem je počítač vybaven. Můžete to zjistit jedním z následujících způsobů:

Přečtěte si *specifikaci* počítače, i když se jen velmi zřídka stává, že se z ní dozvíte dosta-tek spolehlivých informací pro nastavení Linuxu.
Zkuste zjistit, zda je čip FIR připojen jako PCI zařízení. Zadejte příkaz cat /proc/pci. Pou-žíváte-li jádra 2.2.x, správný soubor se jmenuje /proc/bus/pci. Bohužel informace o při-pojených PCI zařízeních nejsou mnohdy vyčerpávající. Aktuální informace o PCI zařízeních a identifikaci výrobců naleznete v dokumentaci jádra v souboru /usr/src/linux/Docu-mentation nebo na stránkách Craiga Harta, *http://members.datafast.net.au/~dft0802/.* Můžete použít také příkaz lspci z balíčku pci-utils.
Vyzkoušejte DOSový nástroj CTPCI330.EXE německého počítačového časopisu CT, ftp://www.heise.de/pub/ct/ctsi/ctpci330.zip. Informace tohoto programu bývají občas přesnější než ty, které zjistí linuxové nástroje.
Zkuste zjistit informace o zařízeních PnP. Užitečným nástrojem může být isapnp.
Pokud máte nainstalovánu podporu IrDA, zkuste nahrávat moduly FIR a sledujte výpis dmesg, zda bude čip FIR detekován.
Další způsob popisuje Thomas Davis: „Navštivte stránky s ovladači výrobce notebooku, najděte ovladač FIR pro Windows 9 a podívejte se na něj. Například pro čip SMC najdete tyto soubory:

-rw-rw-r--1 ratbert ratbert 743 Apr 3 1997 smcirlap.inf -rw-rw-r--1 ratbert ratbert 17021 Mar 24 1997 smcirlap.vxd -rw-rw-r--1 ratbert ratbert 1903 Jul 18 1997 smcser.inf -rw-rw-r--1 ratbert ratbert 31350 Jun 7 1997 smcser.vxd

Pokud máte pochybnosti o identitě čipu, podívejte se do souboru .inf/.vxd pro Windows 95. Windows 95 totiž neobsahují ovladače pro žádné čipy FIR, všechny tedy musí být dodány výrobcem nebo třetími stranami.

7. Thomas Davis také našel balíček dosových nástrojů společnosti SMSC, *http://www.smsc.com/ftpdocs/appsoftware.html.* Balíček obsahuje program FINDCHIP.EXE a také nástroj FIRSETUP.EXE, který by měl být schopen nastavit všechny parametry FIR čipy, s vyjímkou jeho adresy. Kromě toho ještě obsahuje program BIOSDUMP.EXE, jehož výsledkem je následující výstup:

   Příklad 1 (notebook Compaq Armada 1592DT)

   In current devNode: Size = 78 Handle = 14 ID = 0x1105D041 = 'PNP0511' -- Generic IrDA SIR
   Types: Base = 0x07, Sub = 0x00, Interface = 0x02
   Comm. Device, RS-232, 16550-compatible
   Attribute = 0x80

   CAN be disabled

CAN be configured
BOTH Static & Dynamic configuration
Allocated Resource Descriptor Block TAG's:
TAG=0x47, Length=7 I/O Tag, 16-bit Decode
Min=0x03E8, Max=0x03E8
Align=0x00, Range=0x08
TAG=0x22, Length=2 IRQ Tag, Mask=0x0010
TAG=0x79, Length=1 END Tag, Data=0x2F

Zjištěné informace 1:Vstupně-výstupní port – I/O = 0x03E8Přerušení – Irq Tag, Mask = 0x0010 = 0000 0000 0001 0000, jde o binární masku řadiče,

nastaven je 4. bit zprava (počítáno od 0), tedy IRQ 4.

Příklad 2 (neznámý počítač)

In current devNode:Size = 529Handle = 14ID = 0x10F0A34D = 'SMCF010' -- SMC IrCC
Types: Base = 0x07, Sub = 0x00, Interface = 0x02
Comm. Device, RS-232, 16550-compatible
Attribute = 0x80

CAN be disabled
CAN be configured
BOTH Static & Dynamic configuration

Allocated Resource Descriptor Block TAG's:TAG=0x47, Length=7 I/O Tag, 16-bit DecodeMin=0x02F8, Max=0x02F8Align=0x00, Range=0x08TAG=0x22, Length=2 IRQ Tag, Mask=0x0008TAG=0x47, Length=7 I/O Tag, 16-bit DecodeMin=0x02E8, Max=0x02E8Align=0x00, Range=0x08
TAG=0x2A, Length=2 DMA Tag, Mask=0x02, Info=0x08
TAG=0x79, Length=1 END Tag, Data=0x00

Zjištěné informace 2:a) IrCC čip SMC,b) jedna část má I/O 0x02F8, IRQ 3lc) druhá část má I/O 0x02E8, DMA 1

Upozornění

Tento balíček není určen pro koncového uživatele a některé nástroje mohou být potenci-álně nebezpečné. Jediná dokumentace k balíčku je ve formátu Microsoft Word. Uživatelé Linuxu si jej mohou přečíst například pomocí nástroje catdoc na stránkách
http://www.45.free.net/~vitus/software/catdoc/.

Použijte Správce zařízení ve Windows 9x/NT.
Můžete také nahlédnout do dále zmiňovaných informací o hardwaru.
Jako poslední zoufalý pokus můžete vyzkoušet notebook otevřít a podívat se, co je na čipu napsáno.

Informace o hardwaru

Na serveru TuxMobil naleznete informace o IrDA hardwaru, *http://tuxmobil.org/ir_misc.html.* Seznam obsahuje i informace o infračervených zařízeních, o kterých jsme zde nehovořili (myši, tiskárny, dálkové ovladače a podobně).

Aby byl seznam co nejužitečnější, je nutné, aby obsahoval co nejvíce informací o infračervených zařízeních v různém hardwaru. Můžete nám (autorům HOWTO) pomoci zasláním e-mailu s přes-ným názvem hardwaru a typem používaného infračerveného řadiče.

Napište nám také, nakolik vám IrDA podpora v Linuxu fungovala (na jakém tty, portu a přeruše
ní, plus informace o hardwaru – například tiskárnách, mobilních telefonech a podobně). Dále můžete pomoci zasláním podrobných technologických informací o některých infračervených zařízeních, na jejichž základě je možné vyvíjet linuxový ovladač.

Související dokumentace

Linux Infrared HOWTO, *http://tuxmobil.org/howto_linux_infrared.html.*

Přehled konfigurace IrDA IrDA

Podpora infračervených zařízení v linuxovém jádře se velmi rychle vyvíjí. Dále stručně popíšeme způsob jejího nastavení. Podívejte se také do dokumentu Linux Infrared HOWTO, *http:// tuxmobil.org/howto_linux_infrared.html* a na stránky projektu Linux/IrDA, *http://irda.sourceforge.net/.*

Jádro

Potřebujete jádro 2.4 a poslední patche z projektu Linux/IrDA. Jádro 2.6 již podporu IrDA obsahuje.
Zapněte překlad experimentálních částí jádra.
Přeložte jádro s podporou IrDA, sysctl, sériového portu a sítě.

Software

Stáhněte si balíček irda-utils z projektu Linux IrDA.
Rozbalte balíček.
Proveďte make depend; make all; make install.

Hardware

Zapněte podporu IrDA v BIOSu.
Ověřte podporu SIR a FIR výše popsaným postupem.
Příkazem irattach ZAŘÍZENÍ -s 1 zapněte služby IrDA.
Podívejte se na výpis příkazu dmesg.

Linux Infrared Remote Control – LIRC

Balíček LIRC, *http://www.lirc.org/*, podporuje příjem a vysílání infračervených signálů většiny běž-ných infračervených dálkových ovladačů. Obsahuje ovladače zařízení pro hardware připojený k sériovému portu, démona, který prostřednictvím tohoto ovladače dekóduje a vysílá infračerve-né signály, démona myši, který překládá infračervené signály na pohyb myši a několik uživatel-ských programů, které umožňují řídit počítač infračerveným ovladačem. Nakolik tento projekt spo-lupracuje s IrDA zařízeními v noteboocích, nemáme ověřeno.

Čtečka otisků prstů

UPEK, dodavatel oblíbených čteček otisků pro notebooky IBM T42 a další, oznámil, že bude poskytovat knihovnu BioAPI pro biometrickou autentizaci pod Linuxem. Další informace nalez-nete na stránkách *http://linuxbiometrics.com.*

Příslušenství: PCMCIA, USB a další experimentální rozšíření

Karty PCMCIA

Rodiny karet

Ethernetové adaptéry.
Adaptéry TokenRing.
Ethernet + modem/GSM.
Faxmodem/GSM.
SCSI adaptéry.
Vstupně-výstupní karty: RS232, LPT, RS422, RS485, GamePort, IrDA, rádio, video.

Paměťové karty.
Pevné disky.
2,5" diskové adaptéry.

Ke stolním počítačům existují PCMCIA sloty na kartách pro ISA či PCI sběrnici.

Ověření kompatibility s Linuxem

Informace o kartě můžete zjistit příkazem cardctl ident. Pokud v souboru /etc/pcmcia/config není vaše karta uvedena, vhodně modifikujte soubor /etc/pcmcia/<KARTA>.conf. Řiďte se podle obsahu prvního zmiňovaného souboru. Můžete vyzkoušet různé ovladače, třeba karta začne fun-govat – například ovladač pcnet_cs podporuje celou řadu síťových karet PCMCIA kompatibilních s NE2000. Toto řešení není vhodné pro všechny karty, funguje však pro různé neznačkové síťové a modemové karty. Pokud se vám podaří kartu zprovoznit nebo pro ni vytvoříte vlastní ovladač, oznamte to správci projektu PCMCIA-CS Davidu Hindsovi, *http://pcmcia-cs.sourceforge.net/.* Aktuální verzi seznamu podporovaných karet naleznete na adrese *http://pcmcia-cs.sourceforge.net/ftp/SUPPORTED.CARDS.*
V tomto seznamu nejsou uvedeny všechny karty, proto vás odkazujeme také na přehled podpo-rovaných karet PCMCIA/CardBus/CF na serveru TuxMobil, *http://tuxmobil.org/pcmcia_linux.html.*

ExpressCards

ExpressCard je oficiální standard pro modulární rozšíření stolních počítačů a mobilních systémů založený na PCI-Express. Tyto karty představují menší a rychlejší řešení než karty PCMCIA. Na ser-veru TuxMobil naleznete seznam Linuxem podporovaných karet Express Card,
http://tuxmobil.org/expresscard_linux.html.

SmartCards

Informace o čtečkách karet SmartCard naleznete na stránkách projektu Muscle, *http://www. linuxnet.com/smartcard/index.html*, a na serveru TuxMobil, *http://tuxmobil.org/smart_linux.html.*

Karty SDIO

Hledáte linuxové ovladače ke kartám SDIO? V současné době je jich k dispozici naprosté mini-mum, nějaké základní informace naleznete na adrese *http://tuxmobil.org/sdio_linux.html.*

Paměová zařízení – karty RAM a Flash

Projekt Linux Memory Technology Device, *http://www.linux-mtd.infradead.org/*, usiluje o vytvo-ření unifikovaného subsystému pro obsluhu karet RAM a Flash (obecně paměťových zařízení). Usi-luje o kompatibilitu s kódem PCMCIA, aby se tak předešlo duplikaci kódu i úsilí, primárně se nic-méně zaměřuje na malé embedded systémy tak, aby bylo možné ovladač přeložit do jádra a pamě-ťové zařízení použít jako kořenový souborový systém. Velké úsilí je věnováno také minimalizaci velikosti kódu.

Memory Stick

Memory Stick je proprietární paměťové zařízení, používané původně pouze v zařízeních společ-nosti Sony. V současné době jej používají i jiní výrobci. Jde o USB zařízení, která pracují se všemi moderními jádry. Po nahrání modulu usb-storage můžete zařízení připojit jako SCSI disk, typic-ky jako /dev/sda nebo /dev/sdb.

Existuje také adaptér Sony Memory Stick Floppy, MSAC-FD2M. Nevíme ale, nakolik funguje pod Linuxem.

Čtečky karet CD/MMC

Externí čtečky

Všechny externí čtečky SD/MMC/CF karet jsou USB zařízení a bez problémů fungují s modulem usb-storage. Jediný problém může nastat se zjištěním aktuálního přiřazení zařízení. Pomůže vám příkaz dmesg, který by měl po připojení čtečky ukázat, že USB zařízení bylo připojeno například jako disk /dev/sda1.

Interní čtečky

Interní čtečky v současné době existují ve třech provedeních: jako USB, PCMCIA nebo PCI zaří

zení.
USB zařízení jsou méně obvyklá, zpravidla však fungují bez jakýchkoliv potíží. Chovají se stejně jako výše zmíněné externí čtečky.

Některé čtečky jsou zařízení PCMCIA/CardBus. Většinou takovou čtečku najdete poblíž slotu Card-Bus. Takováto zařízení byste měli odhalit příkazem cardctl ident.Na některých noteboocích můžete použít ovladač pro čtečky Winbond W83L518D a W83L519D,
http://mmc.drzeus.cx/wiki/Linux/Drivers/wbsd.

Většina proprietárních zařízení v Linuxu nefunguje. Výjimkou jsou čtečky v PDA Sharp, k nimž existuje binární ovladač s uzavřeným kódem pro platformu ARM.

USB zařízení

Informace o USB zařízeních kompatibilních s Linuxem naleznete na těchto stránkách: *http://www.qbik.ch/usb/devices/* a *http://tuxmobil.org/usb_linux.html.*

Ethernetová zařízení

Podporovány jsou karty AMDtek, D-Link, SMC a další. Podporovány jsou také všechny karty zalo-žené na Pegasus II. Kompletní informace naleznete ve zdrojovém kódu jádra. Pokud máte zaříze-ní s jiným identifikátorem výrobce, než jsou zde uvedené, doplňte je do kódu ovladače a pošlete zprávu na *petkan@dce.bg*.

Klíčenky BlueTooth

Existuje celá řada klíčenek BlueTooth, máme zkušenosti s modely AIRcable, *http://www.aircable.net/*, pro notebooky i PDA (například Sharp Zaurus SL-5x00 a 7x0). Toto zařízení představuje rychlou a jednoduchou metodu, jak mobilní počítač připojit k jinému osobnímu počítači, notebooku nebo mobilnímu telefonu bez použití speciálního kabelu. AIRcable používá spojení technologií Blue-Tooth bez nutnosti složitě BlueTooth nastavovat. Například AIRcable Zaurus-USB lze použít k syn-chronizaci modelů Zaurus (ZaurusManager, Intellisync) s desktopem Qtopia a pro síťová spojení přes jiné PC pomocí pppd. Další informace a přehled kompatibilních zařízení naleznete na serve-ru TuxMobil, *http://tuxmobil.org/bluetooth_linux.html.*

Replikátory portů / dokovací stanice

S těmito zařízeními nemáme žádné zkušenosti, předpokládáme však, že jejich použití pod Linu-xem bude obtížné. Podrobnější informace naleznete v předchozí kapitole.

Tiskárny a skenery

Přehled mobilních tiskáren a skenerů

Přehled portů a protokolů používaných k tisku na mobilních a stacionárních tiskárnách nalezne-te dále v kapitole „Různá prostředí“.

1. Canon, *http://www.canon.com/*, BJC-80 (tuto tiskárnu jde se speciální skenovací hlavou použít i jako skener): David F. Davey píše: „Podařilo se mi rozchodit tiskárnu Canon BJC--80 přes port IrDA. Používám k tomu zařízení pseudo-PostScript, ghostscript a modifiko-vaný lpd:

Jádro linux-2.2.7-ac2-irda6.
/proc/sys/net/irda/slot_timeout zvětšeno na 10 (nutné, jinak neproběhne detekce tiskárny).
ghostscript, DEVICE nastaveno na bjc6000.

- /etc/printcap obsahuje:

```
:xc#01777777:\
:fc#017:\
:fs#020000010002:
```

lpd je opraven tak, aby akceptoval *fs* typu ulong a zpracovával *xc* (v mé verzi lpd nejsou tyto varianty v kódu aktivní, jsou zakomentovány).“
Další informace naleznete na stránce *http://www.windclimber.net/linux/bjc-80.pcgi.*

Tim Auckland napsal: „Pomohla by moje verze lpd? unixlpr je přenosná verze rodiny lpr/lpd kompatibilní s klasickou verzí a s RFC 1179, obsahující několik drobných roz-šíření, obsahuje pole :ms= (které najdete také v SunOS 4) a podporuje přímý tisk na tiskárny připojené přes protokol TCP bez nutnosti použít speciální filtr. ms umožňuje nastavit tty přímo pomocí parametrů pro stty, takže pokud stty zvládá rozšířené pří-znaky, měl by můj lpd fungovat bez dalších úprav.” Poslední verzi unixlpr naleznete na adrese *http://www.geocities.com/CapeCanaveral/Hall/7203/Printing/.*

Canon BJC-50: Zhruba 65 % velikosti BJC-80, součástí jsou Li-Ion baterie, v zásadě stejné funkce jako BJC-80.“
Canon BJ-30.
Citizen, *http://www.citizen-systems.com/*, CN-60.
Pentax, *http://www.pentaxtech.com/*, Pocketjet.
HP DeskJet 340Cbi. Malá přenosná tiskárna. Tiskne buď černě nebo tříbarevně. Občas má problémy se založením papíru. Svou velikostí a přenosností je velmi vhodná pro použití s notebooky. Používali jsme linuxový ovladač HP 500/500C.
Olivetti JP-90.
MaxPoint, *http://www.maxpointgmbh.de/*, TravelScan: mobilní skener připojovaný do portu PCMCIA.

Pokud víme, infračervený port mají pouze tiskárny HP DeskJet 340Cbi a BJC-80. Pokud cestujete do zahraničí, dávejte pozor na napětí v elektrorozvodné síti. Možnost skenování s tiskárnou BJC-80 nemáme ověřenu.

Skenery a OCR software

Projekt SANE, *Scanner Access Now Easy*, *http://www.sane-project.org/*, je API poskytující standar-dizovaný přístup ke všem rastrovým skenerům (stolním, ručním, fotoaparáty, grabbery snímků a podobně). Standard SANE je volně dostupný, jeho vývoj a diskuse nad ním je otevřená. Zdro-jový kód je vytvářen pro Unix (včetně Linuxu) a je uvolněn pod licencí GNU.

Projekt GOCR, *http://jocr.sourceforge.net/*, je software pro rozeznání znaků. Konvertuje soubory
PGM na ASC.Informace o ručních skenerech pod Linuxem naleznete na stránce *http://www.willamowius.de/scanner.html.*

Připojení

Existují různé způsoby připojení tiskárny nebo skeneru k notebooku. Pro tiskárny se obvykle pou-žívají paralelní port, sériový port, USB nebo IrDA. Pro skenery pak paralelní port, SCSI (přes PCMCIA nebo obecný SCSI port), USB nebo PCMCIA. Ke všem způsobům potřebujete odpovída-jící jaderný ovladač.

Sériová zařízení

Sériový adaptér Keyspan PDA

Jednoportový sériový adaptér s konektorem DB-9, dodávaný jako příslušenství k iMacům. Velmi jednoduché zařízení.

Externí úložná zařízení

Externí pevné disky

Existují externí pevné disky s různými způsoby připojení: PCMCIA, USB a FireWire. Dále existují šuplíky pro připojení 2,5" disků (notebooky), 3,5" disků (klasické pevné disky) a 5,25" disků (CD). Všechny v Linuxu fungují bez problémů. Velmi sympatický je šuplík pro 2,5" disk, protože vám umožňuje pořídit do notebooku větší disk a starý disk dále používat jako přídavné zařízení.

Problém: Po probuzení notebooku externí disk většinou nefunguje. Vyřešíte to například odpoje-ním a novým připojením disku nebo disk napájejte externím adaptérem, i když je to řešení poně-kud nepraktické. Disk tak ale zůstane trvale napájen a při uspání notebooku nedojde k jeho vypnutí tak, jako když je napájen z USB portu.

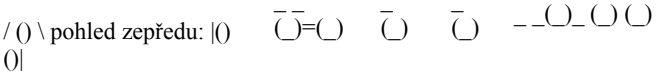
Další problém může nastat s nastavením disku, musí být nastaven jako Master. Většina externích šuplíků nebude fungovat, je-li disk nastaven jako Slave nebo Cable Select.

Napájecí kabely, zdroje

Při cestování do zahraničí byste s sebou měli mít různé typy napájecích šňůr a telefonních kabe-lů. Je také rozumné, aby zdroj dokázal pracovat s různým vstupním napětím, například 110V v USA versus 230V v Evropě. Existují i adaptéry pro napájení 12V v autě.

Některé typy napájecích kabelů:

/ () \ pohled zepředu: |()
()|



označení: C13 C8 ?? PS/2 C5

Upozornění

I když některé konektory mohou být s notebookem fyzicky kompatibilní, dávejte vždy velký pozor na polarizaci a napětí.

Tašky a kufříky

Pravděpodobně vás zajímá, proč zde toto téma vůbec zmiňujeme. Nedlouho poté, co jsem si pořídil Compaq Armada 1592DT, jsem zjistil, že má lehce poškozenou zadní stranu (kde jsou porty pro připojení periferií). Přestože jsem si při převážení notebooku dával velký pozor, poškození bylo způsobeno pokládáním tašky s notebookem na zem. Notebook byl totiž natolik těžký, že si i v tašce sedl přímo na zadní stranu. Proto jsem při převážení notebooku vždy dával na dno tašky měkkou podložku. Pokud s notebookem často cestujete, je vhodná taška nezbytná.

I při převozu v tašce však notebooky bývají často poškozeny. Hlavní příčiny jsou promáčknutí dis-pleje a potlučení rohů. Dobrý kufřík by měl být dostatečně robustní, aby dokázal roznést tlak, a zároveň by měl mít vevnitř dostatek tlumícího materiálu, aby notebooku neublížilo například ani klepnutí s kufříkem o rám dveří.

Mnohem častěji než zničením přijdete o notebook v důsledku krádeže, takže je vhodná i nějaká forma kamufláže.

Exkurze do historie jádra

Tato kapitola ještě není úplně hotová. Obsahuje jen několik poznámek o důležitých změnách mezi jádry 2.4 a 2.6, které se týkají mobilních počítačů. Dále uvádíme několik doporučení k nastavení jádra na notebooku.

Jádra 2.4

Jádra 2.4 se již v moderních distribucích – a na laptopech zvláště – moc nepoužívají. I vy budete pravděpodobně používat novější jádro 2.6.

PCMCIA

Citujeme z PCMCIA.ORG, *http://www.pcmcia.org/*: „PCMCIA (Personal Computer Memory Card International Association) je mezinárodní standardizační výbor a obchodní asociace s více než 200 členy, založená v roce 1989 s cílem vytvořit standardy pro periferní karty a zaručit kompatibilitu mezi mobilními zařízeními, kde jsou základními požadavky odolnost, nízká spotřeba a malé roz-měry. S vývojem potřeb uživatelů mobilních počítačů se změnil i standard PC Card. V roce 1991 navrhla PCMCIA vstupně-výstupní rozhraní se stejným 68pinovým konektorem, používaným původně pro paměťové karty. Ve stejné době byla přidána specifikace Socket Services následova-ná specifikací Card Services. Vývojáři zjistili, že k zajištění kompatibility bude zapotřebí standar-dizovaný software.„ Karty existují ve třech různých formátech: typ I, II a III.

Citujeme ze souboru Documentation/Changes: „V hlavním jádře je nyní částečně implementová-na podpora PCMCIA (PC Card). Pozor při překladu jádra. Pokud budete potřebovat moduly PCMCIA-CS, nepřekládejte jadernou podporu PCMCIA. Pokud nebudete moduly PCMCIA-CS pou-žívat (všechny potřebné ovladače jsou součástí jádra), pak je nepřekládejte. Nezapomeňte také aktualizovat na poslední verzi PCMCIA-CS.“ Další informace naleznete také v souboru README-2.4.

Příklad konfigurace jádra pro notebook naleznete v kapitole „Konfigurace jádra pro notebook“.

Správa napájení

V jádře 2.4 existují dva ovladače pro správu napájení. Každý z nich používá jiné rozhraní do uži
vatelského prostoru: jeden /proc/apm/ a druhý /dev/apmctl/ a /proc/acpi/.Podrobnější informace naleznete na stránkách Johna Fremlina, *http://john.fremlin.de/linux/offbutton/index.html.* Je také autorem programu powermanager.

V jádře 2.4 už je dostupná podpora ACPI, viz dále kapitola o ACPI. Démon Powersave od SuSE sleduje stav baterií, teploty a napájení, umožňuje měnit frekvenci pro-cesoru a nabízí podporu uspání na disk. Podporuje APM i ACPI a dokáže ovládat pokročilé funk-ce řízení spotřeby a hlučnosti pevných disků. Je ideálním programem pro notebooky a pracovní stanice, které musí pracovat potichu a s malou spotřebou, a přitom musí být schopny snadno se přepnout na maximální výkon. Uživatelem definovaná schémata nabízejí plné řízení nad napáje-ním a umožňují snadné a automatické přepínání mezi výkonem a šetřením energie pro každou komponentu zvlášť.

Hotplug

Pro vývojáře zabývajícími se hot-plug vlastnostmi linuxového jádra je k dispozici e-mailová kon-ference na adrese *https://lists.sourceforge.net/lists/listinfo/linux-hotplug-devel.* Týká se zejména zařízení USB, PCMCIA, SCSI a FireWire, neomezuje se však jenom na ně.

Z konfigurační volby jádra:

CONFIG_HOPLUG

Zadejte Y, pokud chcete k počítači za chodu připojovat zařízení a rychle je používat. Ve většině

případů je možné stejná zařízení také odpojit.Známým příkladem jsou karty PCMCIA nebo PC, tedy zařízení jako síťové karty, modemy a pevnédisky připojované do slotů, které jsou k dispozici na všech moderních noteboocích. Dalším pří-kladem využití je sběrnice USB, přítomná na moderních noteboocích i stolních počítačích.

Zapněte HOTPLUG a KMOD a přeložte modulární jádro. Stáhněte si a nainstaluje softwarovéhoagenta. Pak bude jádro automaticky volat user-space agenta, který nahraje moduly a potřebný soft-ware podle toho, jaké zařízení připojíte.

Jádra 2.6

PCMCIA

Balík PCMCIAutils, *http://kernel.org/pub/linux/utils/kernel/pcmcia/pcmcia.html,* obsahuje hot-plug skripty a inicializační nástroje potřebné k tomu, aby se subsystém PCMCIA choval (téměř) stejně jako ostatní hot-plug sběrnice (například USB nebo IEEE1394). Tato podpora byla přidána v jádře 2.6.13-rc1.

Konfigurace jádra pro notebook

Příklad konfigurace jádra 2.4 pro notebook naleznete na adrese *http://tuxmobil.org/kernel_ config_laptop.html.* Nepoužívejte tento konfigurační soubor přímo, vždy konfiguraci vytvořte některým z příkazů make config, make menuconfig nebo make xconfig. Podrobnosti nalez-nete v dokumentu Kernel HOWTO, *http://www.digitalhermit.com/linux/Kernel-Build-HOWTO.html.* Dalším užitečným dokumentem je Linux Kernel HOWTO Thomase Hertwecka, *http://www. thomashertweck.de/kernel.html*, který je však k dispozici pouze v němčině.

Laptopkernel, *http://savannah.nongnu.org/projects/laptopkernel/*, je balíček užitečných jaderných patchů pro uživatele notebooků. Obsahuje podporu ACPI, softwarového uspání, supermount a některé patche zajišťující hardwarovou kompatibilitu. Bohužel od roku 2003 není udržován.

Různá prostředí na cestách

Související dokumentace

Security HOWTO, *http://tldp.org/HOWTO/Security-HOWTO/index.html.*
Multiboot with LILO HOWTO, *http://tldp.org/HOWTO/Multiboot-with-LILO.html.*
Networking HOWTO, *http://tldp.org/HOWTO/NET3-4-HOWTO.html.*
Ethernet HOWTO, *http://tldp.org/HOWTO/Ethernet-HOWTO.html.*
Offline Mailing miniHOWTO, *http://tldp.org/HOWTO/Offline-Mailing.html.*
PLIP HOWTO, *http://tldp.org/HOWTO/PLIP.html.*
SLIP PPP Emulator HOWTO, *http://tldp.org/HOWTO/SLIP-PPP-Emulator/.*

Pokud používáte Debian GNU/Linux, podívejte se do referenční příručky na kapitolu „Network configuration“. Debian obsahuje řadu balíčků, které usnadňují přepínání mezi různými síťovými prostředími.

Konfigurační nástroje

NetEnv

Používáte notebook v různých síťových prostředích? Doma? V kanceláři? U zákazníků?
Pokud ano, mohl by vám pomoci balíček „netenv“. Při bootování vám nabídne jednoduché roz

hraní, ze kterého si můžete vybrat aktuální konfiguraci sítě. Nacházíte-li se v určitém prostředípoprvé, zadáte základní údaje a můžete si je uložit pro příští použití.Netenv vytváří soubor s definicemi proměnných, které popisují aktuální prostředí. Ty lze použít
v konfiguračním schématu PCMCIA, podobně jak to dělá například Debian/GNU a některé dalšídistribuce.Údaje shromážděné pomocí netenv lze použít k nastavení následujících subsystémů:
Síťová zařízení: Nakonfiguruje zařízení podle konkrétního prostředí.
XF86Config: Pokud notebook používáte samostatně s touchpadem anebo připojený k externímu monitoru a myši.
Správce oken: Podle aktuálního umístění počítače si můžete zvolit správce oken.
Tiskové prostředí: Pomocí balíčku netenv můžete snadno nastavit tisk.

Balíček netenv naleznete na adrese *http://netenv.sourceforge.net/*. Ke své činnosti potřebuje dia-log(1). Jeho autorem je Gerd Bavendiek.

SCPM – System Configuration Profile Management

SuSE/openSUSE nabízí software SCPM pro přepínání mezi konfiguračními profily. Můžete nabootovat do jednoho profilu a pak se za běhu přepnout do jiného profilu. Tento software je nástupcem staršího SuSE řešení založeného na správě „schémat“.

Správa profilů v Mandriva Linuxu

Mandriva Linux nabízí – podobně jako openSUSe – ve svém Ovládacího centru možnost nastavení a přepínání mezi různými „profily“. Profily obsahují informace o připojení k síti a též některá další nastavení, viz dokumentaci.

ifplugd

Démon ifplugd, *http:///0pointer.de/lennart/projects/ifplugd/*, je poměrně jednoduchý. Po připojení síťového kabelu nastaví parametry sítě a zruší je po odpojení kabelu. Primárně je určen pro note-booky. Spoléhá na nativní konfigurační subsystém konkrétní distribuce, nepůsobí proto v systé-mu nijak rušivě.

divine

Nástroj divine, *http://www.fefe.de/divine/*, je určen uživatelům, kteří používají počítač v mnoha různých síťových prostředích. Základní myšlenka je:

V souboru /etc/divine.conf popíšete používané sítě včetně jednoho nebo více počíta-čů, které jsou na dané síti s největší pravděpodobností dostupné (vhodnou volbou jsou směrovače nebo servery NIS).
Při nabootování spustíte divine.
Nástroj divine spustí vlákno, které do sítě vysílá ARP požadavky, zkouší to třikrát vždy se sekundovým odstupem. Pokud nedostane odpověď, vlákno vypíše chybové hlášení, skon-čí a ponechá síťové rozhraní v původním nastavení.
Hlavní vlákno pouze čeká na ARP odpovědi a skončí, jakmile nějaká dorazí.
Pro každou síť můžete mít samostatný soubor resolv.conf, například /etc/resolv.conf.default a /etc/resolv.conf.work. Jakmile divine zjistí, v jaké síti jste připojeni, nastaví /etc/resolv.conf jako symbolický odkaz na jeden z vámi defino-vaných souborů.
Pro jednotlivé sítě můžete nastavit server proxy a jeho port, divine tyto údaje zapíše do souboru /etc/proxy. Tyto údaje pak můžete zpracovat některým z inicializačních skriptů shellu, například takto (zsh):

```
export http_proxy="http://`</etc/proxy`/"
```

Součástí programu je perlový skript edit-netscape-proxy, který změní nastavení proxy serveru v konfiguračním souboru prohlížeče Netscape 4.

- Pro každou detekovanou síť můžete doplnit vlastní inicializační skript, který například upraví soubor /etc/printcap či /etc/issue nebo provede cokoliv jiného, co je potře-ba.

Hlavní rozdíl mezi divine a ostatními řešeními spočívá v tom, že ostatní řešení k detekci sítě pou-žívají ping nebo něco

podobného, zatímco divine dokáže velmi rychle otestovat přítomnost v mnoha různých sítích díky tomu, že k detekci sítě využívá ARP dotazy.

Mobilní IP

Z dokumentu Networking HOWTO: „Termínem *IP mobilita* se rozumí schopnost systému přemís-tit své připojení k Internetu z jednoho místa na druhé, aniž by došlo ke změně IP adresy systému nebo ke ztrátě konektivity. Pokud dojde k přepojení systému na jiné místo, se musí za normál-ních okolností změnit i jeho IP adresa. Mobilní IP tento problém řeší tím, že mobilnímu systému přidělí pevnou IP adresu a pomocí zapouzdření (tunelování) IP protokolu a automatického smě-rování zajistí, že datagramy určené pro mobilní systém budou doručeny na tu skutečnou IP adre-su, kterou systém právě používá."

HUT Mobile IP, *http://dynamics.sourceforge.net/*, je dynamický hierarchický systém podpory mobilních IP adres v Linuxu. Implementace umožňuje hierarchický model IP mobility, čímž se minimalizují časy obnovení spojení při změně připojení mobilního systému. Dynamický systém byl navržen s ohledem na technologie Wireless LAN a byl optimalizován právě pro pohyb v těchto sítích.

DHCP/BootP

Protokoly DHCP a BootP jsou rovněž užitečným pomocníkem při používání notebooku v různých prostředích. Další informace naleznete v dokumentu DHCP HOWTO, *http://tldp.org/HOWTO/ DHCP/index.html.*

Volby PPPD

Démona pppd je možné nastavit pomocí více konfiguračních souborů, například pppd file /etc/ppp/<nějaké_nastavení>.

/etc/init.d

Vlastní konfigurační volby můžete definovat přímou editací inicializačních souborů v adresáři /etc/init.d.

PCMCIA – schémata

Jak lze zajistit odlišné nastavení PCMCIA zařízení doma a v práci? Jde to velmi snadno díky podpoře PCMCIA *schémat*. V našem případě použijeme dvě schémata, pojmenovaná například home a work. Podrobnosti naleznete v příslušné kapitole dokumentu PCMCIA HOWTO,
http://pcmcia-cs.sourceforge.net/ftp/doc/PCMCIA-HOWTO.html.

Zavaděče LILO

Následující doporučení pochází od Martina J. Evanse: „První důležitá věc je ta, že jakékoliv para-metry ve tvaru název=hodnota, které nebudou rozeznány jako něco jiného, použije init k nasta-vení proměnných prostředí. Znamená to, že prostřednictvím bootovací výzvy LILO můžete nasta-vovat proměnné prostředí ještě předtím, než se spustí rc skripty. Takto si nastavuji proměnnou LOCATION podle toho, kde počítač zapínám:

LILO: linux LOCATION=home
nebo

LILO: linux LOCATION=work

nebo jednoduše

LILO: linux

přičemž skripty předpokládají, že pokud není proměnná LOCATION nastavena, znamená to nasta-vení LOCATION=home (výchozí nastavení). Abych nemusel při každém bootování ručně zapisovat LOCATION=něco, upravil jsem si soubor /etc/lilo.conf a používám příkaz append:

Linux bootable partition for booting Linux at home # image = /vmlinuz root = /dev/hda3 label = linux read-only # Linux bootable partition config ends # # Linux bootable partition for booting Linux at work # image = /vmlinuz root = /dev/hda3 label = work read-only append="LOCATION=work" # Linux bootable partition config ends

Při bootování doma pak zadávám jenom *linux*, při bootování v práci *work*.Nyní už stačí jenom upravit příslušné inicializační skripty tak, aby nastavení prostředí zohlednilypři volání příkazů jako ifconfig, route a podobných."

Ostatní zavaděče

Kromě LILO existuje i celá řada dalších zavaděčů, například CHooseOS (CHOS) (není licencován pod GPL), GRand Unified Bootloader (GRUB) nebo System Commander. Podívejte se na adresu ftp://metalab.unc.edu/pub/Linux/system/boot/loaders/. Lze

použít i zavaděče z Microsoft Windows NT nebo OS/2.

X Window

Steve, *steve@cygnet.co.uk*, poslal tip na konfiguraci X Window při použití externího monitoru: „Mám-li připojen svůj krásný 17" monitor, spouštím X server bez parametrů a dostanu výchozí 16bitový displej 1152x864. Pokud používám LCD, definuji 15bitový displej (startx --bpp 15) a automaticky dostanu správné rozlišení 800x600. Díky tomu není nutné mít dva konfigurační soubory pro X Window.“

E-Mail

Úvod

Popis, jak nastavit elektronickou poštu na notebooku používaném doma (vytáčené připojení) a v práci (ethernet), zaslal Peter Englmaier (*ppe@pa.uky.edu*):

Vlastnosti

Jako uživatel notebooku mám na nastavení pošty speciální požadavky. Dále popsané nastavení mi umožňuje:

Číst poštu doma na POP serveru poskytovaném univerzitou, stejně tak ji ale mohu číst na počítači v práci.
Odesílat z domu poštu se správnou adresou (která neobsahuje název mého počítače).
Číst a psát poštu na pracovní stanici bez přístupu k notebooku nebo POP serveru (záloha).
Číst poštu na notebooku připojeném do firemní sítě.
Přímý přístup k poště při připojení přes Ethernet (rychlejší než fetchmail).
Nepřímý přístup k poště (přes POP server) při připojení jinak než v práci (doma přes vytáčené připojení nebo úplně někde jinde).
Používat jakýkoliv poštovní program, například elm nebo obyčejný příkaz mail.
Řadit příchozí poštu, mazat spam, rozdělovat souhrnné e-maily do jednotlivých e-mailů.

Konfigurace využívá programů sendmail, fetchmail a vzdáleného účtu na POP serveru.

Konfigurace sendmailu

Tato část je nejsložitější. Mám nainstalovaný balíček sendmail-cf a vytvořil jsem si soubor /usr/lib/sendmail-cf/laptop.mc:

```
divert(-1)
include(`../m4/cf.m4')
define(`confDEF_USER_ID',"8:12")
define(`confBIND_OPTS',`-DNSRCH -DEFNAMES')


# zde definujeme doménu
define(`confDOMAIN_NAME',"pa.uky.edu")
OSTYPE(`linux')
undefine(`UUCP_RELAY')
undefine(`BITNET_RELAY')


# tudy posíláme odchozí poštu
define(`SMART_HOST',`server1.pa.uky.edu')


# tudy posíláme poštu uživatelům, které notebook nezná
define(`LUSER_RELAY',`server1.pa.uky.edu')


# znovu doména, pod kterou chceme být vidět
MASQUERADE_AS(pa.uky.edu)
FEATURE(allmasquerade)
FEATURE(nouucp)
FEATURE(nodns)
FEATURE(nocanonify)
FEATURE(redirect)
FEATURE(always_add_domain)
FEATURE(use_cw_file)
FEATURE(local_procmail)
```

```
MAILER(procmail)
MAILER(smtp)

HACK(check_mail3,`hash -a@JUNK /etc/mail/deny')
HACK(use_ip,`/etc/mail/ip_allow')
HACK(use_names,`/etc/mail/name_allow')
HACK(use_relayto,`/etc/mail/relay_allow')
HACK(check_rcpt4)
HACK(check_relay3)
```

Vypadá to složitěji, než to ve skutečnosti je. Jediné, co touto konfigurací děláme, je, že odchozí poštu směrujeme přes server SMART_HOST a poštu pro neznámé lokální uživatele rovněž pře-směrováváme přes LUSER_RELAY. Díky tomu můžu psát e-mail kolegům bez nutnosti uvádět celou jejich adresu. Důležitější nastavení je, že se hlavička „From“ v odesílaným e-mailech modi-fikuje na záznam MASQUARADE_AS a nebude tak obsahovat přímo adresu notebooku. Bez toho-to nastavení by nemuselo být možné přímo odpovídat na mnou odeslané e-maily. Aby se změny projevily, je nutné sendmail restartovat. Poznámka: Konfigurace je určena pro systém RedHat 5.2, ve svém systému ji možná budete muset změnit.

Nyní je nutné příkazem m4 laptop.mc > /etc/sendmail.cf vygenerovat konfigurační soubor pro sendmail a do souboru /etc/sendmail.cw přidat všechna možná doménová jména, na něž by měl notebook slyšet:

```
# sendmail.cw - sem zadejte všechny aliasy svého počítače laptop laptop.pa.uky.edu
128.17.18.30 guest1 guest1.somewhere.org
```

Je nezbytné, aby v tomto souboru byly uvedeny skutečně všechny aliasy, jinak sendmail poštu nepřijme (a odesílateli odpoví zprávou „we don't relay“). Nakonec musíte celou konfiguraci odzkoušet odesláním pošty a odpovězením na poštu ve všech možných konfiguracích. Každá chyba by mohla vést ke ztrátě pošty.

Konfigurace fetchmailu na notebooku

Jedna z možností, jak zajistit doručení pošty na počítač, je fetchmail. Tento program periodicky vyhledává nové zprávy na jednom nebo více vzdálených serverech. Já používám následující kon-figurační soubor .fetchmailrc uložený v domovském adresáři:

```
set postmaster "myusername" set daemon 900 poll pop.uky.edu with proto POP3 user "mypopusername" there with password "mypoppassword"
is mylaptopusername here
```

Fetchmail nedělá nic jiného, než že převezme nové zprávy a předá je sendmailu, který je umístí do souboru /var/spool/mail/$USER.

Předávání pošty na notebook

Na pracovní stanici mám následující soubor .forward:

```
me@pop.acount.edu,me@server1
```

Server1 je počítač, na kterém mám svou poštovní schránku. Veškerou poštu přeposílám na tento účet, odkud jej posléze (prostřednictvím fetchmailu) vyzvedávám z notebooku. Pokud mám ale notebook připojen k ethernetové síti, chci si poštu číst přímo:

```
me@laptop,me@server1
```

V obou případech se mi pošta zálohuje na *server1* (odkud si ji můžu přečíst v případě, že zrovnanemám po ruce notebook). Veškerou poštu mám uloženou na notebooku.Přepínání řeší tři skripty a crontab na pracovní stanici:Skript forward_pop:

```
#!/bin/sh echo "me@pop.acount.edu,me@server1" > ${HOME}/.forward
```

Skript forward_laptop:

```
#!/bin/sh echo "ppe@laptop,ppe@server1" > ${HOME}/.forward

crontab ${HOME}/mycrontab
${HOME}/utl/check_laptop
```

Skript check_laptop:

```
#!/bin/sh
if /usr/sbin/ping -c 1 laptop                    >/dev/null 2>&1 ; then
:
else
# redirect mail to pop
```

${HOME}/utl/forward_pop

sleep 10

if /usr/sbin/ping -c 1 laptop >/dev/null 2>&1 ; then

back to normal

${HOME}/utl/forward_laptop else # deactivate crontab check /bin/crontab -l | grep -v check_laptop >${HOME}/tmp/mycrontab.tmp
/bin/crontab ${HOME}/tmp/mycrontab.tmp
rm -f ${HOME}/tmp/mycrontab.tmp fi fi

Soubor mycrontab:

mycrontab 0,10,20,30,40,50 * * * * ${HOME}/utl/check_laptop

Při každém připojení notebooku k ethernetové síti spouštím forward_laptop, při každém odpo-jení spouštím forward_pop. Pokud bych zapomněl spustit forward_pop, nejdéle do deseti minut jej za mě spustí crontab. Aby všechno probíhalo automaticky, následujícím způsobem jsem si upravil konfigurační soubory sítě na notebooku:

/sbin/ifdown (skript se spouští při zastavení síťových služeb):

... fi # BEGIN new stuff # turn off forwarding email mail ppe <<EOF turning off forwarding email device = ${DEVICE} hostname = `hostname` EOF if ["${DEVICE}" = "eth0" -a "`hostname`" = "laptop"]; then su -lc "ssh -l myusername server1 utl/forward_pop" myusername >& /dev/null fi # END new stuff

ifconfig ${DEVICE} down exec /etc/sysconfig/network-scripts/ifdown-post $CONFIG

Všimněte si, že skript ověřuje název lokálního systému. Pokud bych totiž byl připojen k nějaké „cizí“ síti, název počítače a IP adresa by byly úplně jiné. /etc/sysconfig/network-scripts/ifup-post (tento skript se spouští po inicializaci síťového zařízení):

Notify programs that have requested notification do_netreport # BEGIN new stuff # check for email -- I'm using fetchmail for this if ["${DEVICE}" = "eth0" -o "${DEVICE}" = "ppp0"]; then su -lc fetchmail myusername >& /dev/null & fi # set clock if connected to ethernet, redirect email if ["${DEVICE}" = "eth0" -a "`hostname`" = "zaphod"]; then (rdate -s server1 ; hwclock --systohc --utc) >& /dev/null & # forward email su -lc "ssh -l myusername gradj utl/forward_laptop" myusername >& /dev/null & fi # END new stuff

exit 0

Zpracování příchozí pošty pomocí procmailu

Toto nastavení je úplně nepovinné. Výše popsaná konfigurace sendmailu spouštěla pro každý při-jatý e-mail procmail, stejně tak byste ale mohli procmail volat prostřednictvím souboru .for-ward (viz manuálová stránka procmail). Procmail je užitečný nástroj na blokování spamu a k tří-dění příchozí pošty.

Abyste mohli procmail používat, potřebujete vytvořit soubor .procmailrc. Podívejte se na man procmail, man procmailrc a man procmailex (příklady). Následující příklad ukazuje, jak některé zprávy ignorovat a jak souhrnné zprávy (digest) rozdělit na jednotlivé části:

-- mail filtering -- procmail is called by sendmail -PATH=/bin:/usr/bin MAILDIR=$HOME/Mail LOGFILE=$MAILDIR/from # keep in mind: # use ":0:" when writing to a file # use ":0" when writing to a device, e.g. /dev/null, or send email

- make a backup of *all* incoming mail, but ignore mail tagged below -:0 c: *! ^Sissa-Repro backup # - keep only last 50 messages :0 ic | cd backup && rm -f dummy `ls -t msg.* | sed -e 1,50d`

- delete email coming through the 'postdocs' email list, when # it is not of any interest :0
* ^From.*postdocs
* ^From.*Ernst Richter /dev/null :0
* ^From.*postdocs
* ^Subject.*card charge /dev/null # Split mailing list from the sissa preprint server into individual emails # - this is quite complicated :(I can flip through the list much
# 1. split it in individual messages :0
* ^From no-reply@xxx.lanl.gov | formail +1 -de -A "Sissa-Repro: true" -s procmail # 2. reformat messages a bit # 2.1. extract 'Title:' from email-Body and add to email-header as 'Subject:' :0 b
* ^Sissa-Repro *! ^Subject TITLE=| formail -xTitle: :0 a |formail -A "Subject: $TITLE " -s procmail
# 2.2. store in my incoming sissa-email folder. Here, we could# also reject (and thereafter delete) uninteresting 'Subjects'# we could also mark more interesting subjects as urgend or send a copy# to regular mail box.:0:

* ^Sissa-Repro
* ^Subject *! ^replaced with sissa

faster and ignore preprints which have uninteresting titles. Instead of

having to browse through the whole list, my mailer will just present a

list of papers.

Mimochodem, ke konfiguraci procmailu existuje nástroj, který se jmenuje ProcBuilder (umí to také Webmin).

E-mail pomocí UUCP

Další možný způsob, jak řešit problémy s e-mailem, představuje protokol UUCP. Tento software byl vyvinut právě pro potřeby počítačů s jen občasným připojením a je to rozhodně nejjednodušší řešení pro notebooky s více uživateli, z nichž každý má vlastní účet.

I když si to řada lidí myslí, UUCP opravdu nepotřebuje připojení přes sériovou linkou, může fun-govat i nad protokolem TCP/IP, takže jako UUCP partnera můžete použít jakýkoliv dostupný počí-tač. Na typickém notebooku bude UUCP soubor sys vypadat takto:
system mylaptop time any chat "" \d\d\r\c ogin: \d\L word: \P address uucp.mypartner.org port TCP

MailSync

Program Mailsync, *http://mailsync.sourceforge.net/*, slouží k synchronizaci více poštovních schrá-nek. Je založen na algoritmu třícestného diffu. Každé dvě schránky se vzájemně porovnávají a zjiš-ťuje se jejich stav při poslední synchronizaci. Poté se mezi nimi zpropagují nově přijaté zprávy a vymazané zprávy. Mailsync dokáže synchronizovat lokální poštovní schránky v různých formá-tech a vzdálené schránky prostřednictvím protokolů IMAP, POP a IMAPS.

Přenos dat mezi různými počítači (synchronizace)

S tímto tématem nemáme praktické zkušenosti. Uvádíme proto jenom přehled různých způsobů přenosu dat a zajištění konzistence dat mezi více počítači.

Užitečný hardware

Externí pevné disky.
Jednotky ZIP.

Wade Hampton napsal: „K přenosu dat můžete použít diskety a ZIP disky naformátované v DOSu. Měli byste také být schopni použít LS120. Pokud máte SCSI rozhraní, můžete použít JAZ, MO nebo i DVD-RAM (jakýkoliv SCSI disk, na nějž můžete zapisovat). Ve svém notebooku Toshiba 700CT mám interní mechaniku ZIP. Funguje skvěle (k připojování používám automount). Na discích používám souborový systém VFAT, takže můžu data přenášet mezi Windows, Linuxem, NT, dávat je kolegům a podobně. Pokud ale potřebuji nahradit interní mechaniku ZIP mechanikou CD, musím notebook vypnout.“

Užitečný software Software pro správu verzí

I když k tomu nejsou primárně určeny, programy pro správu verzí, jako například CVS (Concur-rent Version System), se dají velmi dobře použít v situaci, kdy pracujete na více různých počíta-čích a máte problém s udržením konzistence dat na těchto systémech. Na rozdíl od programů jako rsync, které jsou koncipovány asymetricky (jedna strana je nastavena jako „master“ a vždy pře-píše data na „slave“ stranách), CVS zvládne přijmout změny provedené na kterémkoliv systému a pokusí se je sloučit. Asymetrické systémy jsou vhodné pouze v případě, kdy jste schopni dodr-žovat přísnou disciplínu při práci se soubory nebo pokud přenášíte data z jednoho systému na druhý. Programy jako CVS jsou oproti tomu mnohem tolerantnější.

K synchronizaci dvou nebo více počítačů (typicky pracovní stanice a notebooku) si prostě někde na síti zřiďte CVS repozitář. Může to být na jednom z počítačů, mezi nimiž data synchronizujete, nebo úplně na třetím stroji. Každopádně by měl být tento počítač dobře síťově dostupný a měl by mít kvalitní disky.

Nyní si příkazem cvs update na kterémkoliv počítači pořídíte aktuální verzi sledovaných soubo-rů, můžete je upravovat a příkazem cvs commit změny uložíte do repozitáře. Pokud provedete ve stejném souboru změny na více počítačích, pokusí se je CVS sloučit (většinou tento proces
uspěje), případně vás požádá o ruční vyřešení konfliktu.Typická omezení tohoto řešení: CVS nepracuje dobře s binárními

soubory, takže je vhodnější prouživatele vi nebo emacs než pro uživatele GIMP. Kromě toho má CVS problémy s některými unixovými fintami, jako jsou symbolické odkazy.

Podrobnější informace o CVS naleznete na stránce *http://www.nongnu.org/cvs/*. Dokumentace k systému je na vynikající úrovni.

Souborový systém CODA

Souborový systém CODA, *http://www.coda.cs.cmu.edu/*, je následníkem souborového systému Andrew File System. Stejně jako AFS nabízí CODA transparentní přístup ke sdílenému unixovému souborovému prostoru, který je mapován na několika vyhrazených souborových serverech. CODA je ovšem oproti AFS podstatně vylepšena, takže nabízí mnohem vyšší dostupnost tváří v tvář výpadkům serverů a sítě. Zlepšení dostupnosti bylo dosaženo použitím komplementárních tech-nik replikace serveru a podpory off-line operací. Právě off-line použitelnost je velmi cenná s ohle-dem na podporu notebooků.

unison

Nástroj unison, *http://www.cis.upenn.edu/~bcpierce/unison/*, slouží k synchronizaci souborů mezi Unixem a Windows. Umožňuje uložit dvě repliky sledovaných souborů a adresářů na různých počítačích (nebo na různých discích stejného počítače), kteroukoliv z nich modifikovat a pak pro-vést aktualizaci vzájemnou propagací změn v replikách. Unison byl napsán výzkumnými pracovníky s ohledem na dobře definovanou sémantiku replikací – velký důraz byl kladen na bezpeč-nost a korektní obsluhu událostí jako například předčasné ukončení. Unison je (na rozdíl od rsyn-cu) symetrický/obousměrný, (na rozdíl od cvs) dobře pracuje s binárními soubory a jde o pro-gram uživatelské vrstvy (na rozdíl od mnoha distribuovaných souborových systémů). Rozumně také usiluje o transparentní synchronizaci mezi souborovými systémy Unixu/Linuxu a Windows, což není vůbec jednoduchý úkol. Nevýhoda: Nezajišťuje sledování verzí a neumožňuje synchro-nizaci mezi více než dvěma stromy. Nástroj unison obsahuje řadu funkcí převzatých od systémů pro správu verzí (CVS, PRCS a podobně), distribuovaných souborových systémů (CODA a podobně), jednosměrných zrcadlících nástrojů (rsync a podobně) a jiných synchronizačních programů (Intellisync, Reconcile a podobně).

V celé řadě vlastností se ale liší:

unison běží jak na Microsoft Windows (98, 98, NT, 2k), tak i na Unixu (Solaris, Linux a podobně). (Verzi pro PDA s procesorem ARM naleznete na adrese *http://tuxmobil.org/feed.html.*) Navíc unison pracuje i mezi těmito platformami vzájemně a dokáže tak například synchronizovat notebook s Windows proti unixovému serveru.
Na rozdíl od různých distribuovaných souborových systémů je unison čistě uživatelský program – nevyžaduje žádné úpravy jádra a můžete jej provozovat i bez práv superuživa-tele.
Na rozdíl od jednoduchých systémů pro zrcadlení a zálohování se unison dokáže vyrov-nat s aktualizacemi na obou replikách distribuované struktury. Nekonfliktní aktualizace se propagují automaticky, konfliktní aktualizace se detekují a upozorňuje se na ně.
unison funguje mezi jakoukoliv dvojicí počítačů připojených k Internetu, komunikace je možná přes přímé soketové spojení, tunelováním přes rsh nebo přes šifrované ssh spoje-ní. Pracuje šetrně s přenosovým pásmem a spolehlivě funguje i na pomalých PPP linkách.
unison má jasnou a přesnou specifikaci.
unison je odolný proti selhání. Velmi pečlivě trvale udržuje repliky i vlastní privátní struk-tury ve smysluplném stavu, dokonce i v případech anomálního ukončení nebo výpadku komunikace.
unison je k dispozici zdarma, celý zdrojový kód je k dispozici pod licencí GNU GPL.

TSync

Tsync, *http://sourceforge.net/projects/tsyncd/*, je uživatelský démon zajišťující transparentní syn-chronizaci mezi skupinou počítačů. Používá architekturu peer-to-peer a dosahuje tak vysoké ská-lovatelnosti, efektivity a robustnosti.

MultiSync

MultiSync, *http://multisync.sourceforge.net/*, je otevřený modulární program pro synchronizaci kalendářů, adresářů a dalších dat PIM mezi programy na lokálním počítači a jiných počítačích, mobilních zařízeních, PDA či mobilních telefonech. V současné době MultiSync obsahuje moduly pro kalendáře Ximian Evolution a IrMC Mobile Client (Sony/Ericsson T68i). Synchronizace je možná přes Bluetooth, IrDA nebo kabel.

mirrordir

Mirrordir, *http://linux.maruhn.com/sec/mirrordir.html*, je celá skupina funkcí v jednom balíku. Obsahuje nástroj pro vzdálené přihlášení a démona poskytujícího zabezpečený shell, ekvivalent příkazu cp umožňující navíc kopírovat na servery FTP, nástroj pro lokální nebo FTP zrcadlení sou-borových systémů a další nástroj, jemuž můžete předat skript, který se rekurzivně provede nad soubory.

Mirrordir udržuje přesnou repliku zrcadleného adresáře se všemi detaily pro potřeby časově řízeného zálohování. Kopírují se soubory, u nichž se liší čas modifikace nebo velikost. Duplikují se přístupová práva, vlastnictví, čas modifikace, čas přístupu a sticky bit. Duplikují se zařízení, roury, symbolické i pevné odkazy. Soubory a adresáře odstraněné v hlavním adresáři se v

zrcadlené kopii odstraňují. Automaticky udržuje veškerou podřízenou stromovou strukturu do libovolné hloubky.

InterMezzo

InterMezzo, *http://inter-mezzo.org/*, je nový distribuovaný souborový systém zaměřený na vyso-kou dostupnost. Jde o projekt s otevřeným kódem, jeho cílem je podpora pružné replikace adre-sářů s podporou off-line režimu. Umožňuje například snadnou údržbu kopie domovského adre-sáře na více počítačích a řeší problémy synchronizace stolního počítače a notebooku. Umožňuje replikovat i velká souborová úložiště, a lze jej tak použít k dosažení vysoké dostupnosti serverů. InterMezzo se inspirovalo souborovým systémem CODA, je však navrženo úplně znovu.

WWWsync

WWWsync, *http://www.alfie.demon.co.uk/wwwsync/*, je program v Perlu, který přes FTP aktuali-zuje stránky na webovém serveru podle lokálního počítače. Funguje u jakéhokoliv poskytovatele, který umožňuje přímý FTP přístup k webovým stránkám.

rsync

Program rsync umožňuje kopírovat soubory z a na vzdálený systém velmi podobně, jako to dělá rcp. Oproti rcp má ale mnohem více možností a používá specializovaný protokol, kterým se znač-ně urychluje přenos souborů v případě, že cílový soubor už existuje. Díky tomuto protokolu pře-náší rsync jenom změny mezi dvěma množinami souborů.

Xfiles – synchronizace a validace souborových stromů

Xfiles, *http://www.idiom.com/~zilla/*, je interaktivní nástroj pro porovnávání a slučování dvou sou-borových stromů přes síť. Umožňuje volnou práci na více počítačích (bez zaznamenávání, co se na kterém počítači změnilo). Lze jej použít ke křížové validaci při zálohování. (Některá část disku může být poškozena bez jasné indikace, který soubor je zasažen. Křížová validace proti sekun-dárnímu stromu před samotným zálohováním zajistí, že nezálohujete poškozená data.)

Program s architekturou klient/server (klient je grafický) prochází souborovým stromem a ohla-šuje soubory, které chybí na serveru, na klientovi nebo se na nich liší. Pro každý takový soubor(y) se vypíše velikost a čas modifikace a je možno získat jejich porovnání (unixový diff). Jestliže ve stromu chybí nějaký soubor, jsou ohlášeny *podobně pojmenované* soubory v témže stromě. Nekonzistentní soubory je možné zkopírovat z jednoho stroje na druhý nebo na kterémkoliv ze strojů vymazat. Souborové stromy nemusí být přístupné přes NFS. Paralelně se počítají kontrolní součty souborů, takže i přes pomalou linku je možné rychle porovnat rozsáhlé podobné stromy. Klient a server mohou také běžet na stejném počítači. Pomocí skriptů je možné ošetřit výběr sou-borů a interakci se systémy pro sledování verzí, jako je například RCS. Vyžaduje se Java 1.1 nebo vyšší a JFC/Swing 1.1.

Sitecopy

Sitecopy, *http://www.lyra.org/sitecopy/*, slouží ke kopírování lokálně uložených webových stránek na vzdálené webové servery. Program na server uploaduje lokálně změněné soubory a smaže ze serveru lokálně odstraněné soubory. Jedním příkazem tak udržuje synchronní stav mezi lokálními a vzdálenými stránkami. Jeho hlavním smyslem je eliminace pracného uploadování a mazání jed-notlivých souborů přes FTP.

Konverze dat – adresář, záložky, LDAP, webové stránky

Přenos dat mezi dvěma mobilními zařízeními často vyžaduje nějaký nástroj, kterým se data před importem na cílové zařízení nějak vyextrahují ze zařízení zdrojového – typicky třeba při změně typu mobilního telefonu. Podobná situace nastane, pokud byste chtěli udržovat synchronní adre-sář mezi mobilem a PDA. Nabízíme proto několik odkazů na nástroje pro konverzi záložek (*http://dataconv.org/apps_bookmarks.html*), migraci adresářů (*http://dataconv.org/apps_ addresses.html*), import údajů z formátu vCard (*http://dataconv.org/apps_vcard.html*), slučování LDAP adresářů (*http://dataconv.org/apps_ldap.html*) a konverzi dat mezi PDA a HPC (*http://dataconv.org/apps_pda.html*).

Zálohování

Poškození nebo ztráta dat je na notebooku ještě mnohem pravděpodobnější než na stolním počítači, o to důležitější je tedy zálohování dat. Pro zálohování dat v mobilních prostředích existují různá řešení, podrobněji se jim budeme věnovat v další verzi tohoto dokumentu.

Chcete-li zálohovat na výměnná média jako CD-R/RW nebo DVD-R/RW, můžete nabootovat Knoppix Live CD s volbou toram. Tím se celý Knoppix nahraje do paměti a po nabootování tak můžete jeho CD z mechaniky vyjmout a vložit tak zálohovací médium. (Toto řešení je možné pouze na noteboocích s více než 1GB RAM.)

Připojení k serverům

Dirk Janssen (*dirkj@u.arizona.edu*) napsal: „Nabízím několik užitečných způsobů, jak ze stolního počítače pracovat na notebooku. Pokud máte v práci samostatný stolní počítač, můžete jej použít jako terminálový server notebooku. Tím získáte větší obrazovku a lepší klávesnici bez nutnosti sta-rat se o synchronizaci souborů. Nejjednodušší metodou je mít na obou strojích nainstalováno ssh a pomocí ssh se ze stolního počítače (na němž běží X) přihlásit na notebook; ssh zajišťuje bez-pečné spojení

mezi dvěma počítači, zásadní pro nás je, že zabezpečuje i spojení na X server. Pokud následně spustíte například emacs &, poběží emacs na notebooku, zobrazovat se ale bude v okně na stolním počítači.

Existuje celá řada způsobů, jak zvýšit produktivitu tohoto uspořádání, případně jak je víc zkom-plikovat. Například emacs dokáže příkazem make-frame-on-display otevírat okna (v terminolo-gii emacsu „rámce“) na samostatných displejích. Díky tomu můžete jeden a týž emacs zobrazit jak na stolním počítači, tak na notebooku – ejhle, zrodil se systém se dvěma monitory.

U jiných programů se typicky při spuštění budete muset rozhodnout, na které obrazovce se mají objevit. Chcete-li je zobrazit na obrazovce notebooku, spusťte je obvyklým způsobem. Chcete-li je zobrazit na obrazovce stolního počítače, spusťte je z ssh shellu vytvořeného ze stolního počítače nebo výstup přesměrujte pomocí proměnné DISPLAY. Některé programy dokonce přímo pracují s parametrem -display. Podrobnější informace o nastavení naleznete v dokumentaci příkazu xauth. Jednoduchý způsob, jak zjistit, který pseudodisplej se při ssh přihlášení vytvořil, předsta-vuje příkaz echo $DISPLAY. Řekněme, že stolní počítač se jmenuje velky a notebook maly, pak dostanete něco jako maly:10. Znamená to, že procesy na notebooku zobrazují na to, co považu-jí za lokální displej číslo 10, ve skutečnosti se to ale nějakým ssh kouzlem (zabezpečeně) předá na obrazovku stolního počítače.

Existují i způsoby, jak můžete dynamicky přesunout okno z jednoho počítače na druhý. Zajímavé řešení volí program xmove, jeho uživatelské rozhraní ale není příliš příjemné (dobrovolníci?). Xmove vytvoří pseudoobrazovku (podobně jako maly:10, kterou vytváří ssh) a okna, jejichž výstup je směrován na tuto pseudoobrazovku, pak mohou být přepínána na různé reálné obrazovky (za předpokladu, že všechny mají stejnou barevnou hloubku).

Další možností je spustit některý z programů, které otevřou virtuální kořenové okno – okno na ploše, které obsahuje jiná okna. Trochu se to podobá emulátoru. Pomocí těchto programů může-te spustit aplikaci na jednom počítači, pak okna přesunout na druhý počítač, chvíli s nimi praco-vat tam a potom je třeba zase vrátit na první počítač. Notebook můžete uspat a celý postup opakovat do nekonečna. Podívejte se na programy xmx a VNC.

Pokud vám to přijde příliš složité, nicméně byste chtěli současně používat dvě obrazovky, nain-stalujte si alespoň x2x. Tento drobný nástroj umožňuje přejíždět myší z jedné obrazovky na dru-hou, zároveň se přesouvá i fokus klávesnice. Potřebujete k tomu další ssh spojení z malého (note-book) na velký (stolní počítač), v xtermu na malém zadejte například ssh -Y velky. Shell nechte běžet a příkazem echo $DISPLAY zjistěte, jaká byla vytvořena pseudoobrazovka. Bude to něco jako velky:10 (vysvětlení viz výše). V kterémkoliv shellu na velkém nyní můžete zadat x2x -west -to velky:10 (myslíme tím shell, který beží na velkém a zobrazuje se na velkém, žádný ssh shell). Na levé (západní) straně obrazovky stolního počítače se objeví tenký černý proužek. Jakmile přes něj přejedete myší, přesune se myš na obrazovku velky:10. No a protože to je jenom prostřed-nictvím ssh vytvořený alias obrazovky notebooku, myš se přesune na notebook a můžete na něm psát, aniž byste zvedli ruce od klávesnice stolního počítače.

Poznámka k bezpečnosti systému X Window: Experimenty s různými programy pro manipulaci s obrazovkami budou mnohem snazší, pokud na obou počítačích zadáte příkaz xhost +. Toto nastavení je ale extrémně rizikové, nikdy je nepoužívejte ve větší síti. Jakmile vše nastavíte, dejte si tu práci se zprovozněním xauth. Pokud používáte xdm, je to většinou snadné. V opačném případě zkuste X server spouštět pokaždé se stejným magickým cookie. Je to sice méně bezpečné, ale pořád dostatečně bezpečné a znamená to, že bude muset cookie kopírovat jen jednou. Podí-vejte se do inicializačních skriptů (.xserverrc, .xinitrc, .xsession a podobných) a hledejte něco jako cookie="MIT-MAGIC-COOKIE-1 `keygen`". Tuto hodnotu změňte (vymyslete si vlast-ní cookie): cookie="MIT-MAGIC-COOKIE-1 12345678901234567890abcdefabcdef".“

Bezpečnost v různých prostředích

Úvod

Nejsem odborník na počítačovou bezpečnost, domnívám se nicméně, že zabezpečení mobilních zařízení si vyžaduje zvláštní pozornost. Přečtěte si dokument Security HOWTO, *http://tldp.org/HOWTO/Security-HOWTO/*. V dalším textu uvádíme pouze některé vybrané infor-mace. Popsaná opatření představují pouze doplňkové zabezpečovací mechanismy, i tak vám je ale doporučujeme používat.

Způsoby zabezpečení

Antivirová ochrana: Pro Linux existuje několik antivirových programů. V BIOSu také nastavte zákaz zápisu do bootovacího sektoru.
Notebook jako rizikový nástroj: Protože notebook může být snadno použit k napadení sítě, je rozumné se před připojením notebooku do cizí sítě vždy zeptat jejího správce.
Bezpečné protokoly: Při připojování na vzdálené servery vždy používejte bezpečné proto-koly (například ssh) nebo tunelování (tunnelv, pptp), při přístupu na POP účty použí-vejte protokol APOP.

Ochrana před krádeží

Způsoby ochrany dat

Šifrování: Linuxové jádro nabízí celou řadu možností.
SmartCard: Jediné notebooky s vestavěnou podporou SmartCard jsou Siemens Scenic Mobi-le 800 a některé modely Acer.
Uživatelské heslo: Pokud má útočník fyzický přístup k počítači, lze tuto ochranu snadno obejít.
Heslo pevného disku.
Heslo BIOSu: I tato ochrana se dá poměrně snadno překonat, u notebooků to ale může být složitější než u stolních počítačů. Někteří výrobci (například IBM) umožňují zadat druhé bootovací heslo. Pokud používáte zabezpečení pomocí hesla BIOSu nebo zavadě-če, upozorněte na to! Na kryt notebooku nalepte štítek s textem typu UPOZORNĚNÍ: Tento notebook je chráněn proti krádeži. Heslo může být smazáno jen autorizovaným technikem výrobce proti doložení dokladu o nabytí. Je zbytečné, abyste notebook kradli, nebude vám k ničemu.
Než budete kupovat počítač z druhé ruky, ověřte si, zda není kradený. Odkazy na data-báze kradených notebooků naleznete na adrese *http://tuxmobil.org/stolen_laptops.html.*

Způsoby ochrany hardwaru

1. Zámek notebooku: Většina nových notebooků (ne-li všechny) je vybavena slotem pro zámek. Pokud jej váš notebook nemá, zámky obvykle bývají vybaveny sadou pro namon-tování slotu. Jeden ze zámků Targus Defcon má dokonce detekci pohybu, takže pokud nemáte k dispozici bezpečné místo, nemusíte zámek ani k ničemu zamykat.

   Jediná nevýhoda zámku spočívá v tom, že vybalení (a zamčení) notebooku trvá o chvíli déle. Příprava a zamčení zámku trvá kolem půl minuty a neumožní vám v případě potře-by notebook okamžitě vzít a odejít s ním. Zámek vás také neochrání před připraveným zlo-dějem, který odšroubuje nohu od stolu nebo po kapsách nosí pořádné štípací kleště – pokud si ale potřebujete na chvíli od notebooku odskočit, je to poměrně spolehlivá ochra-na.

   Známí výrobci specializovaných zámků na notebooky jsou Kensignton, *http://www. kensignton.com/*, a Targus, *http://www.targus.com.*

2. Štítek se jménem: Nebezpečí krádeže můžete snížit tím, že si necháte vyrobit štítek se jmé-nem (adresou, telefonem a podobně) a připevníte jej na kryt notebooku. Můžete jej přile-pit oboustrannou lepicí páskou, kvalitní lepidlo je ale mnohem lepší. Při násilném odstra-nění štítku dojde k poškození krytu a notebook je mnohem méně „atraktivní“. Své údaje si můžete také nechat vygravírovat přímo do krytu notebooku (a ideálně na všechny jeho odnímatelné části).
Pokud vám to nevadí z estetického hlediska, notebook si nějakým výrazným způsobem označte, například různými samolepkami. Jednak si jej mnohem snáze poznáte a jednak je takový notebook pro zloděje méně atraktivní.
Propojte služby xlock a apm. Můžete systém nastavit tak, aby se po určité době nečin-nosti neaktivovala standardní apm služba uspání notebooku, ale aby se spustil xlock, vypnul služby apm (tak aby nedošlo k uspání) a spustil „ochranného démona“. Jakmile xlock zmizí, démon se zastaví a služby apm se opět spustí (abyste je mohli vy sami pou-žít).

   Pokud někdo odpojí počítač v době, kdy běží xlock (a nezadá heslo), démon to zjistí a něco udělá – například začne hlasitě přehrávat nějaké volání o pomoc. Kromě toho může démon například pingat (nebo se jinak hlásit) na nějaký lokální server. Pokud se serveru hlásit přestane, aniž by se předem „odregistroval“, server může rovněž provést nějakou akci
   – například poslat SMS, spustit nahrávání kamery v místnosti a podobně. S vypnutými službami apm nebude zloděj schopen snadno notebook umlčet (uspat), protože horké klávesy nebudou fungovat.

V BIOSech Award můžete změnit úvodní logo „pollution preventer“ na něco svého. Návod najdete na stránkách Svena Gegguse, *http://geggus.net/sven/linux-bootlogo.html.* Pro note-booky IBM ThinkPad existuje specializovaný dosový nástroj, kterým můžete na bootovací obrazovku BIOSu naprogramovat své osobní údaje.
Zavaděč je další místo, kde můžete před zavedením operačního systému vypsat své jméno a telefonní číslo (nebo jakékoliv jiné údaje). Takovou „vizitku“ není možné odstranit pro-stou modifikací souboru, a dokonce ani (obyčejným) naformátováním disku. Některé zava-děče umožňují ochranu heslem, kterou vřele doporučujeme zapnout (jinak je totiž velmi snadné získat přístup superuživatele).
Kamufláž – pokud notebook nosíte v brašně na notebook, snáze tak přilákáte pozornost zloděje. Zkuste jej nosit v něčem méně nápadném.
Sériové číslo – na bezpečném místě si poznamenejte sériové číslo notebooku. Může se vám hodit, pokud by vám notebook někdo ukradl.
Pojištění – u některých pojišťoven je možné pojistit notebook proti krádeži.
9. Použijte nějaký program, kterým můžete notebook (i vzdáleně) identifikovat. Kdysi existo-val jakýsi program pro DOS, který dělal něco podobného. Zapsal se do bootovacího sek-toru a při stisknutí určité kombinace kláves vypsal své sériové číslo a zapípal zvukový kód (pro případ, že by obrazovka byla z nějakého důvodu nefunkční). Předpokládalo se, že si sériové číslo své kopie programu zaregistrujete u výrobce.
Notebook se může pokusit po připojení k Internetu odeslat e-mail se svou IP adresou (například z /etc/ppp/ip-up nebo prostřednictvím cronu můžete odeslat e-mailem výpis příkazu ifconfig).
Vždy od notebooku odpojujte všechna externí zařízení a ukládejte je odděleně. Nastav-te BIOS tak, aby prvním bootovacím zařízením byl pevný disk, a vypněte bootování z jiných zařízení. Zdroj se vždy snažte zapínat do co nejhůře přístupné zásuvky, v

případě „bleskové“ krádeže je tak šance, že zloděj nestihne ukrást notebook i se zdrojem a získá tak hůře použitelné zařízení. Elektronické transpondéry – existují zařízení, jejichž polohu je možné vzdáleně detekovat. Informace naleznete na stránkách o krádežích notebooků na serveru TuxMobil,

http://tuxmobil.org/stolen_laptops.html.

Po krádeži

Primární snahou je samozřejmě krádeži notebooku pokud možno zabránit. Pokud k ní ale dojde, měli byste mít k dispozici vše potřebné pro obnovení dat. Nezapomeňte také krádež co nejrychleji ohlásit na polici.

Ošetření nekontinuálního provozu (úlohy cronu)

Program anacron je obdoba cronu, neřídí se ale přesným časem. Je to plánovač periodických úloh, spouštěných v denních intervalech. Na rozdíl od cronu nepočítá s tím, že je počítač trvale zapnutý. Jeho prostřednictvím lze plánovat úlohy spouštěné denně, týdně či měsíčně (nebo v libo-volných jiných n-denních intervalech) na systémech, které neběží 24 hodin denně. Při správné instalaci a nastavení anacron zajistí, že naplánované úlohy budou spuštěny co možná nejpřesně-ji v nastavených intervalech tak, jak to umožní aktuální zapnutí počítače.

Program hc-cron, *http://www.ibiblio.org/pub/Linux/system/daemons/cron/*, je modifikovaná verze široce používaného démona cron Paula Vixieho. Stejně jako původní verze spouští nastavené úlohy v nastavených časech. Originální cron ovšem počítá s tím, že počítač je trvale zapnutý, jinak mohou být naplánované úlohy vynechány. Problém řeší právě hc-cron, určený pro *home-computers*, u nichž se předpokládá, že se běžně vypínají; hc-cron si pamatuje čas vypnutí a dodatečně spustí úlohy, které měly být spuštěny v čase od posledního vypnutí do aktuální-ho zapnutí.

Mobilní tisk

Metod, jak tisknout z mobilních počítačů, je několik. Můžete použít mobilní tiskárnu (viz výše kapitola o tiskárnách a skenerech) nebo můžete tisknout na stacionární tiskárně. Ke stacionární tiskárně nebo tiskovému serveru se můžete připojovat mnoha různými způsoby:
Infračerveně, protokoly IrLPT/IrCOMM, viz dokument InfraRed HOWTO, *http:// tuxmobil.org/howtos.html.*
Infračerveně, protokol IrOBEX, viz dokument InfraRed HOWTO.
BlueTooth, podívejte se na bluetooth backend pro CUPS, *http://www.holtmann.org/ linux/bluetooth/cups.html.* V této chvíli je možný nativní tisk pouze na tiskárny se sério-vým bluetooth portem, plánuje se však i podpora protokolů BPP (Basic Printing) a HCRP (Hardcopy Cable Replacement). Podpora tisku přes Bluetooth je v novějších distribucích součástí balíčku bluez-utils.
Bezdrátová síť, WLAN.
Síť, LAN.
Nástroj rlp, remote line printer.
SMB, Server Message Block, prostřednictvím Samby.
Paralelní port.
Sériový port.
USB port.

Snížení hlučnosti

Vzhledem k rostoucí penetraci mobilními telefony a walkmany není dnes úplně obvyklé starat se o tiché prostředí. Pro ty ohleduplné nicméně uvádíme několik rad.

V počítači vzniká hluk jednak od hardwarových zařízeních (ventilátory, disky) a jednak od provozovaných aplikací.

Konzola (shell) a X Window

Zvuk pípnutí u oken X Window lze zkrátit i snížit příkazem xset b ... (nižší tón většinou působí méně rušivě). Nezávisle na tom lze ve všech oknech kompatibilních s xtermem a v shellu namís-to „akustického zvonku“ nastavit „optický zvonek“. Nastavení setterm -blength 0 v konzole a xset b off v X Window vypíná zvonek. Další podrobnosti naleznete také v dokumentu Visual Bell HOWTO, *http://tldp.org/HOWTO/Visual-Bell.html.*

PCMCIA

Pokud zapnete notebook se správně nastaveným PCMCIA-CS, projeví se aktivace PCMCIA dvojímvysokým pípnutím. Pokud chcete pípání vypnout, do konfiguračního souboru PCMCIA, například/etc/default/pcmcia, přidejte položku CARDMGR_OPTS="-q".

Potlačení akustické signalizace vytáčení na modemových kartách můžete nastavit položkou:

module "serial_cs" opts "do_sound=0"v souboru /etc/pcmcia/config.opts (viz man serial_cs). Tímto nastavením se úplně vypnezvukový výstup, příkazem AT M ale i nadále můžete reproduktor selektivně zapnout a vypnout(například AT M0).

USB

Konfigurační soubor /etc/usbmgr.conf programu usbmgr:

BEEP # beep off # beep on

Hotplug

Do souboru /etc/sysconfig/hotplug přidejte záznam:

HOTPLUG_BEEP="no"

Ventilátory

> Upozornění
>
> Při konfiguraci ventilátorů buďte opatrní. Pokud něco nastavíte špatně, notebook se může přehrát a zničit se. Budete-li chtít chlazení otestovat, vyzkoušejte procesor hodně zatížit, například příkazem md5sum /dev/urandom. Příkazem top byste měli zaznamenat nárůst zatížení procesoru a ventilátor by se měl roztočit rychleji. Pozor, při takovémto testu musíte mít notebook připojen k síti, protože při napájení z baterií procesor obvykle limituje svůj maximální výkon. Sledujte také údaje o teplotě procesoru pomocí příkazu acpi -bt nebo cat /proc/acpi/termal_zone/*.

Pro některé notebooky jsou k dispozici linuxové nástroje pro řízení rychlosti ventilátorů a některých dalších funkcí:

Toshutils, *http://www.buzzard.me.uk/toshiba/index.html*, Jonathana Buzzarda pro některé modely Toshiba.
tpcl, *http://tpctl.sourceforge.net/*, Tomase Hooda pro notebooky IBM ThinkPad.
i8k, *http://people.debian.org/~dz/i8k/*, pro notebooky DELL.

Známé problémy

U některých notebooků je ventilátor trvale, nebo přinejmenším velmi často, zapnutý. Uvádíme několik doporučení, jak to vyřešit.

Snížení frekvence procesoru

V některých případech je ventilátor trvale zapnutý, protože procesor pracuje na příliš vysoké frek-venci. Zkuste ji snížit pomocí nástrojů cpufreqd, *http://sourceforge.net/projects/cpufreqd*, nebo cpu-dyn, *http://mnm.uib.es/gallir/cpudyn/*.

Problémy s přerušením a modulem parport

Někdy způsobuje trvalé zapnutí ventilátoru modul parport. Můžete to vyřešit editací souboru /etc/modules.conf:

alias parport_lowlevel parport_pc options parport_pc io=378 irq=7

Vstupně-výstupní adresa a číslo přerušení závisí na hardwarovém nastavení a/nebo konfiguraci BIOSu. Ve většině případů není nutné číslo přerušení zadávat. O tomto problému a jeho řešení se hovořilo v poštovní konferenci SuSE Laptop, *http://lists.suse.com/archive/suse-laptop/2002 -Nov/0205.html.*

ACPI

V některých případech pomůže změnit nastavení v /proc/acpi/.

Pevný disk

Hlučnost pevného disku můžete snížit stejnými postupy, které jsme uváděli v kapitole o správě napájení. Většina moderních pevných disků v noteboocích je vybavena takzvaným „Acoustic Managementem“, podívejte se do návodu a zjistíte, jak je možné disk nastavit.

Někteří výrobci pevných disků poskytují specializované nástroje. Například Feature Tool, *http://www.hitachigst.com/hdd/support/download.htm*, společnosti Hitachi umožňuje měnit nasta-vení automatického akustického managementu na hodnoty „Lowest acoustic emanation“ (režim Quick Seek) nebo „Maximum performance“ (režim Normal Seek). Některé možnosti správy hluč-nosti nabízí i příkaz hdparm -M.

Různé aplikace

V editoru vi můžete nastavit volbu flash, pak bude chyby indikovat nikoliv zvukem, nýbrž zabli-káním. Do souboru .vimrc nebo v promptu zadejte:

set flash

Případně vyzkoušejte:

set visualbell

Různé využití mobilních počítačů

Úvod

Jak už jsme se dozvěděli, výkon a možnosti notebooků a PDA jsou v některých případech ome-zeny. Na druhé straně mají oproti stolním počítačům výhodu ve své mobilitě. V této kapitole uvá-díme příklady některých aplikací, ve kterých dává smysl použití přenosných počítačů.

Přenosný síový analyzátor

V této oblasti nejsem odborník, zmíním se proto pouze o několika nástrojích, které znám. Kromě klasických příkazů tcpdump či netcat používám ještě další dvě aplikace, které mohou sloužit k analýze síťového provozu.

Multi Router Traffic Grapher (MRTG), *http://www.stat.ee.ethz.ch/mrtg/*, je nástroj pro měření pro-vozu na síťových trasách. Generuje HTML stránky s obrázky, které „živě" reprezentují aktuální zatí-žení linky. MRTG používá Perl a C a funguje v Unixu a Windows NT.

Network Top (ntop), *http://www.ntop.org/*, je unixový nástroj ukazující využití sítě podobně jako oblíbený unixový nástroj top. Je založen na knihovně libpcap a je napsán přenosně, takže může fungovat na prakticky jakékoliv unixové platformě i na Win32. Nástroj ntop lze použít v interak-tivním nebo webovém režimu. V prvním případě vypisuje stav sítě na terminál uživatele. Ve dru-hém případě se k ntopu (který funguje jako webový server) lze připojit prohlížečem. V tomto nastavení se ntop chová v zásadě jako jednoduchý agent protokolu RMON s vestavěným webo-vým rozhraním.

Mobilní směrovač

Projekt *Linux Router Project (LRP)* byl navržen tak, aby nabootoval z jediné diskety, může být uži-tečný i na notebooku.

Průniky do sítí

V souvislosti s možnostmi notebooků může člověka napadnout i problematika průniků do počítačových sítí. Nechceme se zde tomuto tématu věnovat, místo toho vám doporučujeme stránky
http://www.linuxsecurity.com/content/section/9/161/.

Mobilní sběr dat

Související dokumentace

Coffee HOWTO, *http://tldp.org/HOWTO/Coffee.html.*
AX-25 HOWTO, *http://tldp.org/HOWTO/AX25-HOWTO/.*
Serial HOWTO, *http://tldp.org/HOWTO/Serial-HOWTO.html.*
Serial Programming HOWTO, *http://tldp.org/HOWTO/Serial-Programming-HOWTO/.*

Aplikace

Linuxový notebook lze použít ke sběru dat v terénu, například dat geodetických, obchodních, pro-vozních, údajů o pacientech v nemocnicích a podobně. K dispozici je podpora bezdrátového spo-jení přes mobilní telefon nebo amatérské rádio. Nejsme si jisti, zda jsou podporovány radiové karty PCMCIA, podívejte se na stránky Aironet Wireless Communications, *http://www.aironet.com.*

Specifická prostředí

Existují notebooky s mechanickým provedením vhodným do náročných prostředí, existují dokon-ce vodovzdorné notebooky. V některých prostředích (například v nemocnicích) je nutné dát pozor na elektromagnetickou kompatibilitu. Ta může být ovlivněna mnoha faktory, například materiá-lem skříně. Typicky kovové skříně stíní lépe než plastové.

Mobilní kancelář

S programy jako OpenOffice.org, balíky jako KDE (KOffice), *http://www.kde.org/*, Gnome, *http://www.gnome.org/*, a komerčními produkty jako WordPerfect, StarOffice nebo Applixware, *http://www.applix.com/*, je v Linuxu k dispozici celá řada kancelářských aplikací. Společně s odpo-vídajícím hardwarem, například přenosnou tiskárnou a mobilním telefonem, je tak jednoduše možné vytvořit příjemnou mobilní kancelář.

Komunikace s digitálním fotoaparátem

Pokud je nám známo, jsou k dispozici tři způsoby komunikace s digitálním fotoaparátem – infra-červený port (IrDA), sériový port a USB. K dispozici je také řada programů na konverzi obrázků a podobně.

Eric, *dago@tkg.att.ne.jp*, napsal: „Po delší době se mi podařilo stáhnout obrázky z digitálního foto-aparátu, i když ne tak, jak jsem předpokládal (tedy přes USB port), ale pomocí PCMCIA karty a memory sticku, které jsou součástí fotoaparátu. Za zmínku stojí několik drobností:

Sony ukládá obrázky jako soubory JPEG na souborový systém msdos, nejjednodušší způsob, jak je tedy v Linuxu přečíst, je připojit holé zařízení jako souborový systém msdos. Přímo příkaz mount mi nefungoval (nevím proč), s následujícím záznamem souboru /etc/fstab se mi ale podařilo zařízení připojit:

```
/dev/hde1 /mnt/camera msdos user,noauto,ro 0 0
```

Funguje samozřejmě také newfs, ale na tom nic není. Důležité příznaky jsou noauto a ro, bez nich se připojení nezdařilo. Pokud nezadáte příznak ro, odmítne se fotoaparát následně s pamě-tí bavit a musíte ji přeformátovat. Podle dokumentace k fotoaparátu se PCMCIA i USB port cho-vají stejně (na Macu a Windows – uvidíte nově připojený souborový systém). V Linuxu by to prav-děpodobně mělo fungovat také, domnívám se, že kdybych USB zařízení připojil stejně jako to PCMCIA, všechno by fungovalo, jenom se mi nepodařilo zjistit, které zařízení připojit.“

OpenDiS (Open Digita Services), *http://ods.sourceforge.net/*, je knihovna a program pro fotoapa-ráty jako Kodak DC-220, DC-260, DC-265 a DC-250, které používají operační systém Flashpoint Digita. Knihovna je unixovou implementací specifikace Digita Host Interface, jejím účelem je pod-pora systému Digita v dalších produktech, jako je například gPhoto. Samotný program je jedno-duchý řádkový nástroj pro stahování jednotlivých fotografií z fotoaparátu.

Program gPhoto, *http://www.gphoto.org/*, dokáže načíst fotografie z libovolného digitálního fotoa-parátu, vytisknout je, poslat poštou, nahrát na webové stránky, uložit na disk v různých známých grafických formátech nebo prostě jen zobrazit na monitoru. gPhoto obsahuje nový HTML engine, který umožňuje vytvářet tematické galerie (HTML šablony se speciálními tagy), čímž se velmi usnadňuje publikování fotografií na webu. Má implementován i režim prohlížeče adresářů, takže je možné galerie vytvářet i z obrázků již uložených na počítači. Podporuje mimo jiné fotoaparáty Canon PowerShow A50, Kodak DC-240/280 USB nebo Mustek MDC-800. Novější variantou je gPhoto2, kterou využívá i populární program digiKam.

Produkt photopc, *http://www.lightner.net/lightner/bruce/ppc_use.html*, je knihovna a řádkový frontend pro manipulaci s fotoaparáty založenými na čipsetu Fujitsu s firmwarem Siarra Imaging. Program spolupracuje s fotoaparáty Agfa, Epson a Olympus, měl by spolupracovat i se Sanyo. Tyto fotoaparáty se obvykle prodávají s obslužným softwarem pro Mac a Windows, bez popisu komu-nikačního protokolu. Díky tomuto programu s nimi lze pracovat i z unixových systémů. Bruce D. Lightner, *lightner@metaflow.com*, doplnil podporu pro platformy Win32 a DOS. Program nemá žádné grafické rozhraní, dokonce i ve Windows jde pouze o řádkový příkaz. Toužíte-li po grafic-kém prostředí, vyzkoušejte program phototk.

DC20, *http://www.boutell.com/lsm/lsmbyid.cgi/000684*, je balíček pro práci s fotoaparáty Kodak DC20. Obsahuje dva programy, nízkoúrovňový ovladač pro práci z příkazového řádku a TCL/Tk frontend. Můžete použít interní prohlížeč nebo jakýkoliv standardní externí prohlížeč.

kdc2tiff, *http://kdc2tiff.sourceforge.net/*, je program pro konverzi formátu .kdc z digitálních foto-aparátů Kodak DC120 na formát tiff nebo jpg. Program zohledňuje poměr stran, kvalitu, kontrast, gama korekci a otočení obrázku.

rdc2e, *http://freshmeat.net/projects/rdc2e/*, je řádkový nástroj pro stahování fotografií z fotoaparátu Ricoh RDC-2E. K dispozici je ve zdrojové formě nebo jako balíček RPM.fujiplay, *http://packages.debian.org/unstable/graphics/fujiplay.html*, je rozhraní pro práci s digi-tálními fotoaparáty Fuji.

Komunikace s QuickCam (video)

Videokameru lze s notebookem propojit třemi způsoby: přes SV port, FireWire nebo přes USB nevíme ale, nakolik komunikace v Linuxu funguje. Zaslechli jsme i cosi o přenosu videodat přes zvukovou kartu. K dispozici je také dokument Linux QuickCam HOWTO, *http://gentoo-wiki.com/HOWTO_logitech_quickcam_on_2.6.x_kernel.* Program sane, primárně určený pro komunikaci se skenery, by měl podporovat grabování statických obrázků. Program Kino je KDE aplikace, která podporuje kamery připojené přes FireWire rozhraní a umožňuje stahování videa ve formátu DV.

Program kmc_remote, *http://kmc-utils.sourceforge.net/*, je grafické rozhraní pro ovládání digitál-ních kamer Kodak Motion Corder přes sériové rozhraní. Tento program využívá služeb knihovny kmc_serial, součást balíčku kmc_utils. Program nabízí virtuální ovládací panel a sadu příkazových tlačítek pro změnu složitějších nastavení, které se na konzole samotné kamery nastavují více tla-čítky. Funguje podpora všech základních tlačítek, nastavení záznamu (velikost, četnost snímků a podobně) a řízení přehrávání. Jsou podporovány všechny modely kamery ve verzích PAL i NTSC.

Intel PC Camera Pro Pack je jedna z prvních webkamer s USB portem. Tento typ kamer ohlásila i spo-lečnost Sony. Informace naleznete na stránkách *http://www.steves-digicams.com/text_navigator.html.*

Komunikace s televizí

Pokud máte notebook se ZV portem, měl by jít k televizi připojit bez potíží, ať už používáte normu NTSC nebo PAL. Jak je na tom ale podpora v Linuxu, to nám není známo.

Komunikace s mobilním telefonem

Notebook lze s mobilním telefonem propojit přes infračervený port (IrDA), sériový port nebo BlueTooth.

Komunikace s GPS (Global Positioning System)

V dokumentu Hardware HOWTO, *http://tldp.org/HOWTO/Hardware-HOWTO/*, se píše, že pro Linux existuje Trimble Mobile GPS. K jednotce GPS se lze také připojit přes sériový port. Většina GPS přijímačů má datový port a přes speciální sériový kabel dokáže komunikovat s PC.

Diferenciální GPS je technika, při níž se na GPS signál aplikuje korekční faktor vycházejí-cí ze známé polohy. Díky tomu se podstatně redukuje nejistota stanovení polohy. Za nor-málních okolností se korekční signál získává speciálním radiovým přijímačem – dgpsip umožňuje přijímat signál DGPS přes TCP/IP a posílat jej na GPS připojenou k sériovému portu.
DGPS, *http://www.wombat.ie/gps/*, je projekt vytvářející levné hardwarové a softwarové řešení technologie DGPS (a to jak v reálném čase pomocí korekčního formátu RTCM, tak i v režimu následného zpracování).
Balíček gpsd, *http://gpsd.berlios.de/*, je démon, který čte data z přijímače GPS nebo Loran a překládá je do zjednodušeného souřadného formátu, který lze snáze použít v dalších aplikacích. Balík obsahuje i jednoduchého klienta, který vykresluje umístění aktuálně vidi-telných satelitů GPS a měří rychlost pohybu.
Balíček QtGPS, *http://freshmeat.net/releases/28199/*, obsahuje software pro obsluhu GPS přijímače v Unixu/Linuxu/X. Zaznamenává a přehrává trasu, podporuje zobrazení pohyb-livé mapy. QtGPS pracuje se souřadnými systémy délka/šířka a britským OSGB (Ornance Survey).
GRASS (Geographic Resources Analysis Support System), *http://grass.itc.it/*, je volně dostupný rastrový a vektorový GIS, systém pro zpracování obrázků, grafický produkční systém a systém pro prostorové modelování.
XASTIR, *http://www.xastir.org/*, je volně dostupný program pro práci se systémem APRS, Automatic Position Reporting System. APRS byl vyvinut ke sledování mobilních GPS stanic prostřednictvím radiového spojení a umožňuje vytvářet přehledy o poloze, pohybu a podobně. XASTIR tyto informace vykresluje na mapě na obrazovce a umožňuje měnit měřítko zobrazení od celého světa po jednotlivé ulice.
as-gps, *http://www.amphibious.org/gps.html*, je základní knihovna pro obsluhu levných GPS modulů Aisin-Seiki. Balíček také obsahuje několik jednoduchých konzolových nástro-jů pro výpis poloh satelitů, vlastní polohy, času a synchronizaci systémových hodin.
gmap, *http://academy.cas.cz/~gis*, je mapový prohlížeč s důrazem na zobrazení proměn-ných údajů. Jeho vývoj směřuje k volně dostupnému výkonnému systému GIS.
gps3d, *http://www.mgix.com/gps3d/*, je sada nástrojů pro manipulaci s GPS zařízením. Jed-nou z pěkných funkcí je schopnost zobrazit GPS data na OpenGL 3D texturovaném mode-lu země.

Komunikace přes amatérské rádio (HAM)

Pokud vím, notebooky se velmi často používají na soutěžích radioamatérů. Podívejte se na doku
ment HAM-HOWTO, *http://www.radio.org/linux/*, Terryho Dawsona. XASTIR, *http://www.xastir.org/*, je volně dostupný program pro práci se systémem APRS, Automa-tic Position Reporting System. APRS byl vyvinut ke sledování mobilních GPS stanic prostřednic-tvím radiového spojení a umožňuje vytvářet přehledy o poloze, pohybu a podobně. XASTIR tyto informace vykresluje na mapě na obrazovce a umožňuje měnit měřítko zobrazení od celého světa po jednotlivé ulice.

Sledování satelitů

Pomocí antény a programů jako seesat nebo sattrack můžete notebook použít k vyhledávání satelitů na obloze. Při pozorování hvězd můžete použít program xephem. Podívejte se na doku-ment Astronomy HOWTO, *http://tldp.org/HOWTO/Astronomy-HOWTO/.*

Letectví

Řada lidí používá notebook pro různé činnosti související s létáním. Dokument Aviation HOWTO, *http://www.ibiblio.org/fplan/Aviation-HOWTO/*, obsahuje odkazy na různé linuxové programy, které mohou být užitečné pro soukromé, komerční nebo armádní piloty. Snahou samozřejmě je umožnit pilotům veškeré své počítačové potřeby realizovat výhradně na systému Linux.

Nevidomí a zrakově postižení

Jsou skupiny lidí, pro které může notebook představovat velmi užitečného pomocníka. Příkladem takové skupiny mohou být osoby nevidomé nebo zrakově postižené. Podrobnější informace naleznete v dokumentu Accessibility HOWTO, *http://tldp.org/HOWTO/Accessibility-HOWTO/*, a na stránkách projektu Blinux (Linux for blind people), *http://leb.net/blinux/.* Program brltty podpo-ruje různé Braillovy terminály. Program festival je systém pro syntézu řeči. K dispozici jsou pro-gramy pro zvětšení části obrazovky a kurzoru. Další informace na téma instalace Linuxu pro nevi-domé uživatele naleznete na serveru TuxMobil na adrese *http://tuxmobil.org/mobile_blind.html.* Pro české uživatele budou určitě zajímavé stránky *http://www.brailcom.org/.*

Ostatní operační systémy

Microsoft DOS a Windows

Úvod

Mohou nastat důvody, proč mít na notebooku nainstalován jak Linux, tak i Microsoft DOS/Win-dows. Často není pro Linux k dispozici podpora flash ROM nebo některých PCMCIA karet či modemů nebo chcete ve Windows zjistit hardwarové informace, které nejsou v důsledku nedo-statečné podpory výrobců v Linuxu viditelné. Ne všechny tyto úkony lze provést v emulovaném prostředí jako DOS-EMU, WINE nebo VMware.

Pokud budete chtít Linux s X Window, Netscape a podobně a zároveň Windows 9x/NT/2000/XP, disk menší než 1 GB bude trochu těsný. I tak se nám ale podobná instalace podařila na disku
o velikosti 810 MB.

Dosové nástroje pro změnu rozložení disku

Notebook často koupíte s předinstalovanou kopií Microsoft Windows. Pokud byste chtěli jen zmenšit diskový oddíl s Windows, potřebujete nějaký nástroj, který dokáže měnit velikosti oddílů
– dnes to již umí instalační programy některých distribucí (openSUSE, Mandriva Linux). Druhá varianta samozřejmě je oddíl nejprve odstranit, vytvořit nové oddíly a pak systém znovu nainsta-lovat. Většina následujících informací pochází ze stránek Michaela Egana, *Michael.Egan@sonoma.edu.*

Známý a spolehlivý je komerční program Partition Magic, *http://www.powerquest.com/product/pm/index.html*, společnosti Power Quest.Sharewarový program BootitNG, *http://www.bootitng.com/*, dokáže měnit velikost diskových oddí-lů se souborovými systémy NTFS, EXT2, EXT3 a ReiserFS.

System Commander 2000 společnosti Symantec dokáže změnit velikost diskového oddílu se sou-borovým systémem FAT32. Na rozdíl od Partition Magicu dokáže SC2000 fungovat i bez nainsta-lovaného operačního systému společnosti Microsoft (ovšem i

Partition Magic se dá použít ze dvou samostatných disket).
Jedním z „novějších“ nástrojů pro přerozdělení disku a změnu velikosti oddílů je Ranish Partition Manager, *http://www.ranish.com/part/.* Podporuje FAT32 a pracuje se i na podpoře linuxových souborových systémů.

Řada lidí ke změně velikosti diskových oddílů se souborovým systémem FAT používá program FIPS 15c, *http://bmrc.berkeley.edu/people/chaffee/fips/fips.html.* Jinou verzí z jiného zdroje je FIPS 2.0, *http://www.igd.fhg.de/~aschaefe/fips/.*

Sdílení diskových oddílů

Linux a Windows spolu mohou sdílet odkládací oddíl. Podrobnosti naleznete v kapitole „Systémy s omezenými prostředky“. V Linuxu můžete připojit jakýkoliv DOS/Windows oddíl se souborovými systémy msdos, vfat, nebo dokonce komprimované oddíly (jako Drivespace a podobně). Chcete-li používat dlouhé názvy souborů, zvolte typ vfat, a pokud chcete provádět i automatickou konverzi (velmi užiteč-né u textových souborů), můžete nastavit volbu conv=auto. Ve svém /etc/fstab používám následující nastavení, za určitých okolností ale může vést k podivnému chování – podrobnosti viz dokumentace jádra.

/dev/hda8 /dos/d vfat user,exec,nosuid,nodev,conv=auto 0 2

Řešení z opačného konce umožňují nástroje na adrese *http://uranus.it.swin.edu.au/~jn/linux/*, které ve Windows 9x/NT umožňují pracovat se souborovým systémem ext2.Dalším podobným nástrojem je LREAD, *http://www.it.fht-esslingen.de/~zimmerma/software/ltools.html*, pro Windows 9x/NT (a dokonce pro DOS a Windows 3.x), který rovněž umožňuje při-stupovat k souborovému systému ext2. Stejnou funkcí disponuje i oblíbený Total Commander. Pomocí těchto nástrojů můžete vypisovat obsah adresářů a kopírovat soubory z Linuxu do DOSu

a naopak. Můžete také mazat soubory a modifikovat jejich přístupová práva.Balík dále obsahuje jednoduchý server, který umožňuje přistupovat k souborům ze vzdálenýchklientů po síti (jedná se ale o bezpečnostní riziko, protože řízení přístupu je v tomto případě velmijednoduché).

LINE Is Not an Emulator

LINE, *http://line.sourceforge.net/*, umožňuje ve Windows spouštět linuxové binární soubory tím, že zachytává linuxová systémová volání. Běh samotných aplikací se neemuluje, běží přímo na pro-cesoru stejně jako nativní aplikace Windows. Nezdá se ale, že by byl v použitelném stádiu.

Instalace bez CD mechaniky

Můžete použít CD mechaniku stolního počítače (nebo zkopírovat obsah CD na pevný disk) a pak oba počítače propojit null-modemovým kabelem. Komunikaci mezi počítači zajistíte programem INTERLNK.EXE, který můžete spustit po nabootování z dosové diskety.

Různé

Program Laplink, *http://www.travsoft.com/.*Windows NT obsahuje službu RAS, Remote Access Service.Windows 9x/NT podporují protokol PPTP, jímž lze vzdálená zařízení propojit TCP/IP tunelem.

Tento protokol je podporován i v Linuxu. Pro Linux existuje PPTP server PoPToP,

http://www.poptop.org/, který umožňuje transparentní komunikaci linuxových serverů v prostředí PPTP VPN. Administrátor sítě tak může využít výhodné kombinace klienta Microsoft a serveru Linux. Současná verze serveru je plně kompatibilní s IETF standardem PPTP a podporuje klienty na Windows i na Linuxu s plným rozsahem šifrovacích a autentizačních možností.

BSD Unix

FreeBSD je unixový operační systém pro platformu PC. Má vlastní podporu PCMCIA zařízení, APM a dalších oblastí, souvisejících s mobilními počítači.

PicoBSD, *http://people.freebsd.org/~picobsd/*, je jednodisketová verze FreeBSD 3.0, která v různých variantách umožňuje bezpečný vytáčený přístup, malý bezdiskový router, nebo dokonce server telefonického přístupu – to vše na jediné disketě 1,44 MB. Ke spuštění mu stačí 386SX a 8MB RAM, pevný disk není nutný. Tuto distribuci lze použít také k instalaci BSD na notebooku, podobně jako se používají linuxové mikrodistribuce.
PAO, FreeBSD Mobile Computing Package, *http://www.jp.freebsd.org/PAO/.*

Projekt CMU Monarch, *http://www.monarch.cs.cmu.edu/*, implementuje Mobile-IPv4 a Mobile-IPv6 na platformě FreeBSD.
XF86Config Archive, *http://www.sanpei.org/Laptop-X/Laptop-X/*, je archiv různých souborů XF86Config, používaných uživateli Linuxu a FreeBSD. Pokud hledáte XF86Config pro svůj notebook, podívejte se sem.
Pokud víme, FreeBSD již podporuje IrDA.
Poštovní konference FreeBSD-Mobile, *http://lists.freebsd.org/mailman/listinfo/freebsd-mobile.*
Informace o instalaci FreeBSD na různé typy notebooků naleznete na adrese

http://www.jp.freebsd.org/PAO/LAPTOP_SURVEY/.

OS/2

Stránky Notebook/2, *http://www.os2warp.be/index2.php?name=notebook2,* dr. Martinuse obsahují informace o různých noteboocích a PCMCIA kartách, které pod OS/2 fungují.

Novell Netware

Klientská strana v operačních systémech DOS/Windows nepředstavuje žádné problémy, protože většina PCMCIA karet má i ovladač pro Netware. Chcete-li se připojovat z Linuxu, bude vás zají-mat balíček mars_nwe. Propracovanou podporu Novellu má také distribuce Caldera Linux.

Prozatím jsme nezkoušeli instalovat na notebook Netware server, takže nevíme, jestli existují i serverové ovladače PCMCIA karet.

Debian GNU/Hurd (hurd-i386)

GNU Hurd je nové jádro operačního systému vyvíjené FSF. Hurd je tou poslední komponentou, díky níž je možné sestavit úplný GNU operační systém – jedním z takových je například Debian GNU/Hurd. Současná verze projektu je určena pro architekturu i386, budou však podporovány i další.
Příručka GNU Hurd Hardware Compatibility Guide, *http://www.nongnu.org/thug/gnumach_ hardware.html*, říká, že Hurd funguje i na noteboocích, prozatím ale nemá funkční podporu PCMCIA.

Další zdroje informací

Důležité WWW stránky

Kenneth E. Harker udržuje velmi hodnotnou databázi „Linux on Laptops“, *http://www.linux-on-laptops.com/.* Na těchto stránkách naleznete aktuální informace o poštovních konferencích, dis-kusních skupinách, časopisech a celé řadě webových stránek, které se týkají problematiky note-booků.

Autor příručky provozuje stránky věnované problematice instalace Linuxu na různých noteboo-cích, *http://tuxmobil.org/mylaptops.html*, společně s informacemi o různých hardwarových kom-ponentách.

Poštovní konference

Přehled důležitých poštovních konferencí, některé z nich jsou převzaty z Kennetových stránek.

Obecné konference

Do konference *Linux-Laptop-Mailing-List* na serveru TuxMobil se můžete přihlásit na stránce *http://tuxmobil.org/mobilix_ml.html.* Naleznete tam také archiv konference. Jde o poměrně novou konferenci, má už však dostatečně silnou uživatelskou základnu.

Do konference *Linux-Laptop-Mailing-List* na serveru Kernel.Org se můžete přihlásit zasláním e-mailu s předmětem subscribe linux-laptop na adresu *majordomo@vget.kernel.org.* Dostanete potvrzující zprávu, na kterou musíte správně odpovědět. Původně tato konference fungovala na adrese *majordomo@vger.rutgers.edu* a šlo o konferenci se značným provozem. Současný provoz je mnohem nižší, takže se zdá, že se změnou adresy konference ztratila velkou část uživatelů.

Konference věnované konkrétním distribucím

K dispozici je konference *debian-laptop*, která se zabývá otázkami a diskusemi ohledně provozo-vání operačního systému Debian GNU/Linux na noteboocích. Přihlásit se můžete zasláním e-mailu s předmětem subscribe na adresu *debian-laptop-request@lists.debian.org.* Můžete také navštívit webové stránky *http://www.debian.org/MailingLists/subscribe* a přihlásit se prostřednictvím for-muláře. Konference se archivuje, má rozumný objem provozu a dobrou kvalitu.

openSUSE rovněž nabízí konferenci *suse-laptop*, *http://www.suse.de/de/private/support/online_help/mailinglists/index.html*, která probíhá převážně v němčině.Nejsou nám známy žádné české nebo slovenské specializované konference o noteboocích sezaměřením na konkrétní distribuci. Většina problémů se řeší v hlavních (a jediných) konferencích,jimiž jsou *linux@suse.cz* nebo *mandrake@mandrake.cz*, případně přímo v obecnél*inux@linux.cz.* Podobná situace je i se specializovanými diskusními skupinami.

Konference věnované konkrétním notebookům

Konference *linux-dell-laptops*, *http://tech.groups.yahoo.com/group/linux-dell-laptops/*, je věnována problematice Linuxu na noteboocích DELL. Má minimální provoz a vlastní archiv.Konference *linux-thinkpad* je věnována problematice Linuxu na IBM ThinkPadu. Její provoz jeminimální. Přihlásit se můžete na adrese *majordomo@bm-soft.com.* Další konferencí věnovanou Linuxu a Thinkpadu je *linux-thinkpad* na adrese *http://lists.topica.com/lists/linux-thinkpad/.* Má rovněž minimální provoz.Konference *linux-toshiba-portege*, *http://groups.yahoo.com/subscribe.cgi/linux-on-portege*, jevěnována Linuxu na Toshiba Portege. Má minimální provoz a vlastní archiv.

Konference *linux-tosh-40xx* je věnována Linuxu na Toshiba Satellite 40xx. Má minimální provoz. Přihlásit se můžete na adrese *majordomo@geekstuff.co.uk.*

Diskusní skupiny

Diskusní skupiny USENETu představují zdroj informací o různých tématech souvisejících s provo-zováním Linuxu na noteboocích, která často ještě nebyla nikde popsána. Pokud se vám nedaří nalézt potřebné informace v tomto dokumentu ani v žádném z těch, na které se odkazujeme, doporučujeme vám zeptat se na vhodné diskusní skupině.

Linuxové diskusní skupiny

comp.os.linux.portable,
comp.sys.mac.potrables,
comp.os.linux.answers,
comp.os.linux.hardware,
comp.os.linux.misc,
comp.os.linux.setup,
cz.comp.linux,
cz.comp.linux.debian,
cz.comp.linux.mandrake,
cz.comp.linux,suse.

PDA – diskuse a IRC kanály

comp.sys.handhelds,
comp.sys.palmtops,
comp.sys.pen,
#zaurus@irc.freenode.net.
irc.freenode.net #opie #opie.de.

Diskusní skupiny kolem X Window

comp.windows.x,
comp.windows.x.apps,
comp.windows.x.i386unix.

Hardwarové diskusní skupiny

comp.sys.laptops,
alt.periphs.pcmcia,
comp.sys.ibm.pc.hardware.chips,
comp.sys.ib.pc.misc.

RSS kanály

- TuxMobil News (RDF/RSS), *http://tuxmobil.org/newsfeed.html*, je dostupný i ve formě měsíčních souhrnných e-mailů.

Časopisy

Časopisy a magazíny obecně o PC a noteboocích, mobilních počítačích, Unixu nebo Linuxu.

pcLaptop Magazine, *http://www.pclaptops.com/.*
Mobile Computing and Communications, *http://www.mobilecomputing.com/.*
Road Warrior News, *http://warrior.com/rwsub1.html.*
PCMCIA Update Newsletter, *http://www.pc-card.com/pcmcia8.htm.*
The Linux Journal, *http://www.linuxjournal.com.*
UNIX Review, *http://www.unixreview.com/.*
iX Multiuser Multitasking Magazin (v němčině), *http://www.heise.de/ix/.*
Computer Shopper, *http://www.zdnet.com/~cshopper/.*
Linux+, *http://www.lpmagazine.org/.*
LinuxEXPRES, *http://www.linuxexpres.cz.*

Obecné informace o noteboocích

Následující stránky obsahují informace o noteboocích obecně, bez ohledu na používaný operač
ní systém.Laptop Soup, *http://www.laptopworldwide.com/laptops.html*, nabízí celou řadu informací o tom,jaké společnosti vyrábějí jaké počítače pod různými značkami. Pokud vás zajímá, která společnostvyrobila váš notebook, nejspíš to zjistíte právě zde.

The WWW Virtual Library: Handheld Computing, *http://handheld.teco.edu/*, obsahuje obrovskouspoustu informací o časopisech, konferencích, akademických projektech a dalších tématech.Najdete zde také odkazy na různé standardy, neziskové a vládní organizace či výrobce.

Federal Communications Commission On-line Equipment Authorization Database,*http://www.fcc.gov/fcc-bin/ead.* Pokud máte problém s identifikací výrobce svého notebooku(nebo jiného elektronického zařízení), můžete na těchto stránkách nalézt FCC ID v databázi FCC.Tento identifikátor naleznete na všech výrobcích určených pro Spojené státy.

Oprava hardwaru

Mohou nastat různé důvody, proč notebook či PDA rozmontovat.

Oprava vadného hardwareu
Zjištění jinak nedostupných informací o hardwaru (například z popisu nedetekovaného čipsetu).
Odstranění reproduktorů („speakerektomie“, popsaná v dokumentu Visual Bell HOWTO. *http://tldp.org/HOWTO/Visual-Bell.html*).
Flash BIOSu (obvykle na základní desce).
Výměna baterie BIOSu.
Upgrade pevného disku.
Upgrade paměti.
Instalace dalšího hardwaru, například interní bezdrátové miniPCI karty.

Pokud notebook není v záruce, jeho oprava může být poměrně drahá, v některých případech navíc chybí profesionální podpora. I samotné otevření notebooku však může být složité. Postupy pro upgrade paměti nebo disku bývají popsány v návodu. Další podrobnosti zjistíte v servis-ním/technickém manuálu. Postupujte velmi opatrně a značte si, kam který šroubek patří. Všech-ny musíte vrátit do těch správných dírek, jinak by mohlo dojít k poškození základní desky a zni-čení notebooku. I po odšroubování všech šroubků (a některé bývají dobře schované) jednotlivé části obvykle drží pohromadě díky nalepeným plastovým držákům, takže je musíte opatrně oddě-lit. V některých případech budete potřebovat speciální nářadí (zejména různé exotické šroubová-ky) nebo páječku. Hodně štěstí.

> Upozornění
>
> Výrobci notebooků a PDA typicky podmiňují záruku tím, že do zařízení mohou zasahovat jen profesionální servisní pracoviště. Pokud se do tohoto chcete pustit sami, pak možná naleznete zajímavé informace na stránkách *http://repair4laptop.org/*, *http://repair4pda.org/*, *http://repair4mobilephone.org/*, nebo dokonce *http://repair4printer.org/.*

Informace o linuxových mikrodistribucích

Linuxové mikrodistribuce jsou vzhledem ke svým minimálním nárokům velmi vhodné pro note-booky – zejména máte-li například služební počítač s nainstalovaným systémem Microsoft Win-dows. Jsou také vhodné při instalaci s pomocí nelinuxového počítače. Existuje několik mikrodi-stribucí, které nabootují z jedné či dvou disket nebo CD/DVD.

Další informace naleznete například na stránkách *http://www.linuxhq.com/*, případně *http://tinux.sourceforge.net/.* Užitečný dokument je také BootDisk HOWTO na adrese *http://tldp.org/HOWTO/Bootdisk-HOWTO/index.html.* Následující odkazy a tipy

shromáždil Matt-hew D. Franz, vývojář Trinuxu. Další podobný přehled konzolových a minidistribucí nabízí Fresh-Meat, *http://ma.us.mirrors.freshmeat.net/appindex/console/mini-distributions.html.*

1. Knoppix, *http://www.knopper.net/knoppix/index-en.html*, Klause Knoppera je bootovatel-né CD s výběrem linuxového softwaru, automatickou detekcí hardwaru a podporou mnoha grafických karet, zvukových karet, SCSI a USB zařízení a dalších periferií. Knoppix lze pou-žít jako demonstrační a výukové CD, jako záchranný disk nebo jako platformu k demon-straci komerčních produktů. Není nutné nic instalovat na pevný disk. Díky komprimaci je na jednom CD umístěno téměř 2 GB spustitelných programů. K dispozici je také kix (Knop-pix mini CD).
Vlastní live-CD nebo live-DVD má dnes téměř každá distribuce – Mandriva Linux, Fedora Core, openSUSE a další. Specialitou z českých končin je distribuce Slax, která se vejde i na mini CD nebo USB flashdisk.
MuLinux, *http://mulinux.dotsrc.org/*, Michele Andreoliho.
tomsrbt, *http://www.toms.net/~toehser/rb/*, Toma Oehsera, GNU/Linux na jediné disketě použitelný jako distribuční nebo záchranný disk.
Trinux, *http://www.trinux.org/*, Matthewa D. Franze, soubor bezpečnostních nástrojů.
LRP, Linux Router Project, *https://www.psychosis.com:4301/linux-router/.*
hal91, *http://jspiro.tripod.com/linux/hal91.htm.*
floppyfw, *http://www.zelow.no/floppyfw/, Thomase Lundquista.*

C-RAMDISK, *http://www.ibiblio.org/pub/Linux/kernel/images/.*
PocketLinux, *http://www.pocket-linux.org/.*
SmallLinux, *http://www.superant.com/smalllinux/*, Stevena Gibsona. Třídisketová minidi-stribuce. Kořenový disk používá souborový systém ext2 a obsahuje programy fdisk a mkfs.ext2, takže je možné začít instalovat na pevný disk. Vhodný k nabootování star-ších počítačů s méně než 4 MB RAM.
Linux-lite, *http://www.ibiblio.org/pub/Linux/kernel/*, Paula Gortmakera pro velmi malé systémy s méně než 2 MB RAM a 10 MB diskového prostoru (jádro 1.x.x).
Podívejte se také na balíčky na adrese http://www.ibiblio.org/pub/Linux/system/ recovery/!INDEX.html.
Do této kategorie spadají také různé bootovací diskety jednotlivých distribucí, například bootovací a záchranná disketa distribuce Debian GNU/Linux.
Pokud si chcete vytvořit vlastní bootovací disketu, můžete to udělat ručně postupem popsa-ným v dokumentu Boot Disk HOWTO, *http://tldp.org/HOWTO/Bootdisk-HOWTO/index.html*, nebo můžete použít některý z nástrojů jako mkrboot nebo pcinitrd.
Můžete také vytvořit linuxový systém na disku ZIP. Postup naleznete v dokumentu ZIP Install HOWTO, *http://tldp.org/HOWTO/ZIP-Install.html.*

Systémy s omezenými prostředky, vyladění systému

Související dokumentace

LBX HOWTO, *http://www.tldp.org/HOWTO/LBX.html.*
Small Memory HOWTO, *http://tldp.org/HOWTO/Small-Memory/.*
Lightweight Linux, část 1, *http://www-128.ibm.com/developerworks/linux/library/l-lwl1/*: Hardware je starý jenom tak jako software, který na něm běží. Moderní operační systém a aktuální verze aplikací umožní produkční využití i starých systémů. Tento článek popi-suje vhodné metody a podrobný návod, jak vytvořit funkční linuxový systém na starším hardwaru nebo i na moderním hardwaru s omezenou velikostí paměti a disku.

Úvod

Jak už bylo řečeno v úvodní kapitole, ve srovnání se stolními počítači mohou být prostředky na noteboocích omezené. Tato kapitola by vám měla pomoci vyrovnat se s omezeným prostorem, pamětí, výkonem procesoru a baterií.

Málo diskového prostoru Úvod

Existuje několik různých technik, jak získat více diskového prostoru, například sdílení disku, uvolnění nepoužívaného nebo redundantního prostoru, vyladění souborového systému a komprimace.

> Upozornění
>
> Některé techniky šetří diskový prostor na úkor větších paměťových nároků. Jak uvidíte, k uvolnění místa na disku vede řada drobných kroků. Většinu z nich má smysl používat pouze u opravdu velmi malých disků v řádech stovek megabajtů, maximálně gigabajtů, nebo tam, kde máte k dispozici velmi omezené zdroje (flash disky, různé paměťové karty a podobně).

Techniky

1. Stripping – i když většina dnešních distribucí obsahuje již stripnuté binární soubory, není na škodu to ověřit. Podrobnosti

zjistíte příkazem man strip. Jednotlivé nestripnuté sou-bory naleznete buď příkazem find nebo pohodlněji příkazem findstrip.

Upozornění

Nestripujte knihovny, v důsledku nevhodných programátorských technik by mohlo dojít k odstranění nesprávných symbolů. Můžete také použít volbu --strip-unneeded.

Perforace – příkaz zum(1) čte seznam souborů ze standardního vstupu a pokouší se tyto soubory perforovat. Perforací se rozumí, že sekvence nulových bajtů budou nahrazeny operací lseek, takže souborový systém má možnost pro tyto oblasti souborů nealokovat skutečný fyzický prostor (detaily v kapitole „Souborové systémy“ z části „Příručka správce operačního systému“). Příklad: find . -type f | xargs zum.
Odstranění nepotřebných a duplicitních souborů – zkuste v systému najít a smazat core soubory, recovery soubory emacsu (*#soubor#*), recovery soubory vi (*soubor.swp*), recove-ry soubory RPM (*soubor.rpmorig*) a recovery soubory nástroje patch. Ke hledání duplicit-ních souborů můžete použít příkaz finddup. Zvolte si vhodný systém k pojmenovávání záloh, dočasných a testovacích souborů.
Odstranění dočasných souborů – typicky v adresáři /tmp, můžete použít i nástroj tmp-watch.
Zkraťte logovací soubory – typicky v adresáři /var/log. Můžete k tomu použít nástroj log-rotate.
Odstranění souborů – odstraňte soubory, které nejsou „nezbytné“ – například různé manu-álové stránky, dokumentace v /usr/doc a zdrojové soubory v /usr/src.
Nepotřebné knihovny – balíčkem binstats můžete nalézt nepoužívané knihovny.
Souborový systém – zvolte si takový souborový systém, který s diskovým prostorem pra-cuje úsporně, například rsfs. Zkuste svůj souborový systém vyladit například nástrojem tune2fs. Zvolte vhodné velikosti oddílů a bloků.
Zmenšení velikosti jádra – můžete jednak odstranit nepotřebné funkce jádra, jednak pou-žít komprimovaný obraz bzImage.
Komprimace – souborový systém můžete zkomprimovat příkazem gzip a pak jej dekom-primovat za běhu. Druhá možnost spočívá v komprimaci jen vybraných souborů. Kompri-mované binární soubory můžete spouštět prostřednictvím příkazu zexec.
11. Komprimované souborové systémy – pro souborové systémy e2fs je k dispozici kompri
movaná verze e2compr, *http://e2compr.sourceforge.net/*.Pokud používáte komprimované souborové systémy ve Windows (drivespace, doublestacker),můžete k nim přistupovat pomocí nástroje dmsdos, *http://cmp.felk.cvut.cz/~pisa/dmsdos/*.
Sdílení diskových oddílů – mezi různými souborovými systémy můžete sdílet odkládací oddíl (viz *http://tldp.org/HOWTO/Swap-Space.html*) i datové oddíly (viz mount). Pro při-pojení komprimovaných dosových oddílů můžete použít program dmsdos.
Knihovny – použijte jiné (starší) knihovny, například libc5, která je menší než libc6, známá též jako glibc2.
Jádro – pokud vaše potřeby splní i starší jádra, použijte je, jsou menší.
GUI – co nejvíce se vyhýbejte grafickým rozhraním.
Útlé distribuce – Existuje několik distribucí, které se vejdou od jedné 3,5" diskety do 10 MB diskového prostoru a nepotřebují ani mnoho paměti. Viz též přílohy „Ostatní operační systémy“ a „Informace o linuxových mikrodistribucích“ nebo další text v této příloze.
Externí úložná zařízení (pevné disky, ZIP, NFS, SAMBA) – vzhledem k tomu, že rozšiřitel-nost notebooků je obvykle omezena, atraktivní volbou je paralelní port. Pro ten jsou k dis-pozici jak externí pevné disky, tak mechaniky ZIP. Lze je obvykle také připojit přes PCMCIA. Další možnost je prostřednictvím sítě a služeb jako NFS či SAMBA použít pro-středky jiného počítače.
Smazání nepotřebných lokalizačních dat – localepurge je jednoduchý skript, který smaže nepotřebné lokalizační soubory a lokalizované manuálové stránky. V závislosti na instala-ci je tímto nástrojem možné ušetřit 200, 300 nebo i více megabajtů diskového prostoru, zabraného lokalizačními údaji, které pravděpodobně nikdy nebudete potřebovat.

Rychlost pevného disku

Pomocí nástroje hdparm můžete dosáhnout lepšího výkonu pevného disku. Než jej vyzkoušíte, v BIOSu zkontrolujte nastavení parametrů disku, jako je DMA, ATA4 nebo 32bitový přenos. Špat-né je, že pokud některou z těchto funkcí BIOS nepodporuje, pomocí hdparmu ji nebudete moci zapnout.

V dokumentu Tunable Filesystem Parameters in /proc, *http://www.diverge.org/ulcj/199910tfsp.shtml*, se dozvíte, jak vyladit chování souborového systému.

Málo paměti Související dokumentace

Small Memory HOWTO, *http://tldp.org/HOWTO/Small-Memory/index.html.*
Module HOWTO, *http://tldp.org/HOWTO/Module-HOWTO/.*
Kerneld HOWTO, *http://tldp.org/HOWTO/Kerneld/.*

Techniky

Zkontrolujte využití paměti nástroji jako free a top.Projekt Mergemem, *http://www.complang.tuwien.ac.at/ulrich/mergemem/.*
Řada programů obsa-huje paměťové oblasti se stejným obsahem, aniž by to operační systém detekoval. Typicky tytooblasti obsahují data vygenerovaná při spuštění, která se dlouhodobě nemění. Mergemem doká-že takové oblasti najít a sdílet. Sdílení se

odehrává na úrovni operačního systému a programy nauživatelské úrovni si ho nejsou vědomy. Mergemem je užitečný, zejména pokud spouštíte více
instancí interpretů a emulátorů (jako jsou Java či Prolog), které svůj kód ukládají do privátníchdatových oblastí. S menší úspěšností však tato technika funguje i na jiné programy.Dále můžete co možná nejvíce omezit velikost jádra: jednak odstraněním všech pro vás nepo-
třebných funkcí jádra a jednak co nejvyšší mírou modularizace.

Ukončete také všechny nepotřebné služby, například lpd, mountd, nfsd, a zavřete některé vir-tuální konzoly. Další podrobnosti naleznete v dokumentu Small Memory HOWTO.Je-li to možné, můžete samozřejmě použít odkládací prostor.Pokud to jde, zkuste využít prostředků jiného počítače – například pomocí X11, VNC nebo pro-
stě jen telnetu nebo ssh. Podrobnosti o VNC naleznete na adrese *http://www.realvnc.com/*.

Nízká rychlost procesoru

Můžete sice zkusit procesor přetaktovat, ale hrozí zde nebezpečí poškození hardwaru. Některé pří-klady naleznete například na adrese *http://www.silverace.com/libretto/*.

Techniky snížení spotřeby

Pokud nepotřebujete infračervený port, vypněte jej v BIOSu nebo zastavte ovladač zařízení IrDA. Některé funkce IrDA ovladače lze využít ke snížení spotřeby.
Služby PCMCIA mají velkou spotřebu, takže pokud je nepotřebujete, vypněte je.
Svou spotřebu má i podsvícení displeje.

> Upozornění
>
> Podsvícení má omezenou životnost ve smyslu cyklů zapnutí/vypnutí. Vyhněte se proto spořičům obrazovky, které podsvícení vypínají.Pokud přesto chcete podsvícení vypínat, můžete použít příkazy xset +dpms a xset dpms0 0 300. Tím se po pěti minutách neaktivity vypne obrazovka. Funguje to jen v případě,že monitor podporuje DPMS.

Prodloužení doby běhu lze dosáhnout i úpravou baterií, viz *http://repair4laptop.org/ notebook_battery.html.*
Informace o APM naleznete v samostatné kapitole.
Pokud souborový systém připojíte s volbou noatime, nebude jádro aktualizovat údaj

> o čase posledního přístupu k souboru. Tato informace je sice občas užitečná, většině uži-vatelů je ale k ničemu. Její zápis můžete bez problémů vypnout, čímž předejdete zápisu na disk při každém čtení souboru. Příklad souboru /etc/fstab s touto volbou vypadá takto:
> /dev/hda7 /var ext2 defaults,noatime 0 2

Nástrojem hdparm můžete nastavit zastavení disku a další parametry. Podporuje i některé možnosti SCSI.
Démon Mobile Update, *http://www.complang.tuwien.ac.at/ulrich/linux/tips.html*, je náhra-dou standardního démona update. Minimalizuje roztáčení disku a manipulaci s diskem tím, že obsah bufferů zapisuje na disk jen v případě, že došlo k jiné diskové aktivitě. Kon-zistenci souborového systému vynutíte ručním zadáním příkazu sync. Jinak by mohlo dojít ke ztrátě dat například při výpadku napájení. Démon mobile-update nepoužívá služeb APM a funguje tak i na starších systémech.
Nástroj noflushd, *http://noflushd.sourceforge.net/*, sleduje diskovou aktivitu a po nastavené době nečinnosti disk zastaví. Je vhodné jej kombinovat s nástrojem hdparm a s připojením oddílů s volbou noatime. Dále uvádíme komentáře a postřehy Nata Makareviče o možných způsobech redukce diskové aktivity v Linuxu. Užitečným zdrojem informací je soubor Docu-mentation/filesystems/proc.txt, zejména pasáž /proc/sys/vm. S jádrem 2.2 můžete použít:

```
echo "100 5000 8 256 500 60000 60000 1884 2" > /proc/sys/vm/bdflush
```

> U jader 2.4, která ve volném čase předukládají méně používané stránky paměti do odklá-dacího souboru, dochází ke zvýšení diskové aktivity. Mně se osvědčilo (jádro 2.4.9, 192 MB RAM, notebook Toshiba 3480) následující nastavení. Pozor: Některé parametry mohou být nebezpečné nebo zbytečné (nemám k dispozici seriozní data o praktické účinnosti nasta-vení). Obecně platí, že jakékoliv opožďování diskových zápisů je z podstaty riskantní.

```
echo 99 512 32 512 0 300000 60 0 0 > /proc/sys/vm/bdflush
# is '60' the max value for age_super?
echo 1 1 96 > /proc/sys/vm/buffermem
echo 512 128 32 > /proc/sys/vm/kswapd
echo 1 10 96 > /proc/sys/vm/pagecache
```

Toshiba Linux Utilities, *http://www.buzzard.me.uk/toshiba/index.html*, je sada nástrojů pro řízení ventilátoru, nastavování hesel a

funkcí horkých kláves na noteboocích Toshiba. Lze použít také KDE balíček *Klibreta*.
Kenneth E. Harker na svých stránkách doporučuje program LCDProc, *http://lcdproc. omnipotent.net/*. Jde o malý program, který umožňuje na LCD displeji 20x4 zobrazovat základní informace o systému. Mimo jiné zobrazuje stav baterií u notebooku. Program jsme zkoušeli a zjistili jsme, že komunikuje pouze s externím displejem Matrix-Orbital, *http://www.matrixorbital.com/*, který se připojuje přes sériový port. Pro notebooky je to pravděpodobně k ničemu.
Démon Diald, *http://diald.sourceforge.net/*, podle potřeby zajišťuje vytáčené připojení pomocí protokolů SLIP nebo PPP. Dokáže se v případě potřeby automaticky spojit se vzdá-leným systémem a ukončí neaktivní spojení.
KDE obsahuje balíčky KAPM, Kbatmon a Kcmlaptot. Balíček Kcmlaptop Paula Campbella je sada ovládacích panelů, které implementují podpůrné funkce pro notebooky, například zobrazuje aktuální stav baterie a upozorní vás, jakmile je baterie příliš vybita. Dovoluje také nastavit parametry řízení spotřeby. Podobné balíčky naleznete i v projektu GNOME.
Další informace můžete nalézt také v dokumentu Battery Powered HOWTO,

http://tldp.org/HOWTO/Battery-Powered/.

Ještě několik slov k zastavení pevného disku pomocí nástrojů noflushd nebo hdparm. Smyslem je samozřejmě minimalizovat používání pevného disku, protože ten v notebooku představuje hlav-ní zdroj hluku a jeden z hlavních spotřebičů energie. Démon noflushd je náhradou démona update, který zapisuje obsah diskových bufferů na disk jen tehdy, pokud dochází ke čtení nějа-kých dat z disku (oproti tomu démon update se chová tak, že buffery ukládá každých 5 sekund, a obvykle tak generuje konstantní diskový provoz, kvůli němuž nikdy nedojde k zastavení disku). Démon noflushd také nastavuje prodlevu pro zastavení disku a před samotným zastavením volá funkci sync. Používá se například jako „noflushd -n 5 /dev/hda“. Použití tohoto démona může vést ke ztrátě dat v případě, že upravujete nějaký soubor při zastaveném disku a neproběhne sync

– například v důsledku výpadku napájení.Nástrojem hdparm je rovněž možné nastavit čas uspání disku a také vyladit výkon disku pomo-
cí některých parametrů. Zkontrolujte, zda máte v jádře v části „Block devices“ zapnutu volbu „Use DMA by default when available“. Samotné zapnutí noflushd nebo nastavení uspávání disku ve většině případů nestačí, protože ve standardní instalaci v systému běží řada úloh cronu, zapisuje se do logů, používá se odkládací soubor a podobně. Tyto činnosti nejsou vždy nutné, zejména pokud počítač není připojen k síti a používá jej jenom jeden uživatel. Dále uvádíme několik doporučení.

Ze systému můžete odstranit démony cron a jemu podobné (anacron, atd, logrotate) v případě, že nepotřebujete jimi spouštěné služby (například mazání dočasných adresářů a logů, kontrola pošty a podobně).
Dále můžete upravit konfigurační soubor /etc/syslog.conf a omezit počet logovacích souborůa logovaných událostí, kromě toho před názvy jednotlivých souborů můžete uvést znak „-“, kterýříká, že není nutné po zapsání každého záznamu provést sync na disk.

Dále můžete do konfigurace syslogu přidat záznam „mark:none“, takže se nebudou zhruba každou půlhodinu zapisovat zprávy o tom, že syslogd běží. Logy typické linuxové instalace jsou dnes pro normálního domácího uživatele zbytečně obsáhlé.

A konečně, k uspání disku nedojde, pokud se pracuje s odkládacím souborem. Zadejte příkaz„free“ a zjistíte, nakolik se používá odkládací soubor a kolik je k dispozici paměti RAM. Pokudmáte k dispozici dostatek RAM, takže se nejspíš obejdete bez odkládacího souboru, nebo pokudje využito hodně odkládacího souboru a zároveň je hodně volné paměti, můžete zkusit odkláda-cí prostor uvolnit (su; swapoff -a; swapon -a) nebo úplně vypnout (su; swapoff -a). Nasystémech s více než 64 MB paměti byste měli být schopni se obejít bez odkládacího souboru[2].(Pokud vypnete odkládací soubor, samozřejmě se zmenší velikost dostupné paměti a některé pro-gramy mohou v případě nedostatku paměti bez varování zhavarovat. Máte-li ovšem nějaký pro-gram s enormní spotřebou paměti, nezabrání mu ve zhroucení ani odkládací soubor, ten dokážehavárii jen pozdržet.)

S popsanými opatřeními byste měli být schopni po delší dobu pracovat s vypnutým pevným dis

kem.Jádro je možné nastavit s podporou APM a s vypínáním konzoly pomocí APM. V konzolovém reži-mu se tak bude automaticky vypínat monitor (obrazovka nejen zčerná, ale vypne se i její podsví-cení). V grafickém režimu můžete stejného efektu dosáhnout příkazy xset +dpms (zapne DPMS)a xset s blank (aktivuje vypínání obrazovky). Tyto příkazy můžete přidat do inicializačníchskriptů grafické relace nebo správce oken.

Volby řízení spotřeby v BIOSu (uspávání disku, zhasínání monitoru a podobně) nejsou obvyklek ničemu a v některých případech mohou dokonce způsobit havárii počítače. Proto je lepší tytovolby v BIOSu vypnout.

Jádro Související dokumentace

Kernel HOWTO, *http://www.digitalhermit.com/linux/Kernel-Build-HOWTO.html.*
BootPrompt HOWTO, *http://tldp.org/HOWTO/BootPrompt-HOWTO.html.*

S notebooky souvisí celá řada funkcí jádra, například APM, IrDA, PCMCIA a některé specializova-né volby pro konkrétní notebooky, například IBM ThinkPad. V některých distribucích nejsou tyto volby ve výchozím nastavení zapnuty, jádro také bývá větší, než je nutné. Je proto rozumné pře-ložit si vlastní jádro. I když to pro začátečníka může být obtížnější, rozhodně je to vhodné. Nebu-deme se této problematice podrobněji věnovat, protože je popsána v mnoha jiných dokumentech.

Přeložte si modulární jádro s moduly pro CDROM, disketu, PCMCIA, zvukovou kartu a další peri-ferie. Pak budete mít možnost odložit zavedení příslušných modulů až na dobu, kdy jsou skuteč-ně zapotřebí, a systém se také snáze zotaví z hardwarových problémů (například vadného CD disku), protože máte možnost modul odstranit a znovu zavést bez nutnosti restartovat systém.

[2] Poznámka českého vydavatele: Údaj o velikosti paměti je poněkud zastaralý a na vašem počítači bude zcela jistě vyšší. Vypnutí odkládacího prostoru bych doporučoval pouze v případě, že dobře víte, co děláte.

Štíhlé aplikace a distribuce

Prozatím jen malá sbírka, stále shromažďujeme další informace. BOA – „Lightweight and High Performance Web Server“. Jde o jednoúlohový HTTP server. To zna-mená, že na rozdíl od klasických webových serverů neprobíhá fork pro každé nové spojení ani se nevytváří více kopií k obsluze současných spojení. Server interně multiplexuje mezi všemi pro-bíhajícími spojeními a k forku dochází jen při volání CGI programů (které musí běžet jako samo-statné procesy). Předběžné testy ukazují, že BOA zvládá na Pentiu 100 obsluhovat několik stovek požadavků za sekundu.

MGR – grafický okenní systém, který potřebuje mnohem méně prostředků než X Window.
Low Bandwidth X – Alan Cox na konferenci Linux Redux v únoru 1998: „...Existují dva pro-jekty, které dobře zvládají normální aplikace. LBX (Low Bandwidth X) je oficiální aplikací X konsorcia, Dxpc je alternativa, kterou preferuje více uživatelů. Tyto systémy pracují jako proxy X servery a u normálních požadavků komprimují datový tok dobře o 50 procent, často se jim daří snížit původní datový tok až na 25 procent. S programem dxpc jsou apli-kace X Window rozumně použitelné i přes modemovou linku 28,8 nebo přes Internet.“
Blackbox, *http://blackboxwm.sourceforge.net/* – správce oken pro X. V řadě ohledů je podobný populárním balíkům jako Window Maker, Enlightenment nebo FVWM2. Tento balíček vás bude zajímat, pokud vás nebaví správci oken s obrovskými nároky na systé-mové prostředky, chcete však používat atraktivní a moderně vyhlížející prostředí. Příbuz-ným Blackboxu je Fluxbox (*http://fluxbox.sourceforge.net/*) s většími možnostmi nastave-ní.
XFCE, *http://www.xfce.org/index.php*, je nenáročné a stabilní prostředí pro různé unixové systémy.
Linux-lite – distribuce založená na jádru 1.x.x, vyžaduje pouze 2 MB paměti a 10 MB diskového prostoru.
SmallLinux, *http://www.superant.com/smalllinux/*, je třídisková mikrodistribuce Linuxu, založená na jádře 1.2.11. Používá souborový systém ext2 a obsahuje nástroje fdisk a mkfs.ext2, takže je možné instalovat i na pevný disk. Lze ji použít k nabootování na starších počítačích s méně než 4 MB RAM.
cLIeNUX – linuxová distribuce orientovaná na používání v klientském režimu.
Minix, *http://www.minix3.org/* – nejde o Linux, ale o UNIX pro velmi malé systémy, napří-klad procesory 286 a 640 KB RAM. Existuje dokonce i podpora X11 pojmenovaná mini-x, ftp://ftp.linux.org.uk/pub/linux/alan/.
Screen – malý, ale mocný správce konzoly. John M. Fisk (*fiskjm@ctrvax.vanderbilt.edu*) napsal v Linux Gazette: „Žijeme ve světě GUI, GUI, GUI! Nebo se vás o tom hlavní výrob-ci operačních systémů alespoň snaží přesvědčit. Pravda je ale taková, že i když se grafic-ká prostředí používají hodně, jsou situace, kdy řádkové rozhraní stále představuje velmi efektivní způsob práce. Je rychlé, efektivní a vhodné pro počítače s omezenou pamětí nebo výkonem procesoru. I v konzole se dá udělat spousta šikovných věcí.“
Tinyirc – malý IRC klient. Nemá většinu pokročilých příkazů jako klienti z rodiny ircII, není ani barevný, ale funguje a je malý.
JOVE – Jonathans Own Version of Emacs, malý, ale výkonný editor.

Upgrade hardwaru

Můžete se pokusit upgradovat hardware svého notebooku, nemusí to ale být úplně jednoduché. Pokud vás zajímají různé možnosti, podívejte se na stránky *http://repair4laptop.org/*.

Notebooky a ekologie

Srovnání počítačů z ekologického hlediska

V rámci projektu ReUse, *http://www.reuse-computer.de/*, na Technické univerzitě v Berlíně, *http://www.tu-berlin.de/*, odborníci zkoumali spotřebu energie v různých fázích životního cyklu počítačů. K výrobě počítače je zapotřebí 535 kWh, což je asi o 10 % méně než před čtyřmi lety. Většina energie je spotřebována při používání počítače, například osm hodin denně. Spotřeba nových počítačů s procesory 2,5–3 GHz je i v klidovém režimu stále okolo 100 W, počítače s pro-cesory 1,4 GHz potřebují 80 W a čtyři roky starý počítač potřeboval jen 60 W. Z ekologického hle-diska je tedy vhodnější pořídit si starší počítač, k jehož vyrobení už

není zapotřebí žádná další energie a jeho spotřeba při používání je rovněž nižší.

LCD displeje mají menší spotřebu než klasické monitory, notebooky proto z výzkumu vycházejí jako nejekologičtější počítače, při používání mají nejmenší nároky na energii. Starší notebooky jsou opět lepší, protože jejich procesory mají nižší spotřebu.

Ekologickými aspekty se zabývá Ecology HOWTO v páté části knihy.

Grafické čipsety NeoMagic série NM20xx

Úvod

Grafické čipsety NeoMagic série NM20xx byly oblíbené v noteboocích vyráběných kolem roku 1996. Po dlouhou dobu byly tyto čipsety podporované jen v komerčních X serverech. V polovině roku 1998 uvolnil RedHat binární server X11 společnosti PrecisionInsight. Od verze 3.3.3 tento čip-set podporuje i XFree86/X.org.

Textový režim 100x37

Tato kapitola je z velké části dílem Cedrica Adjiha. Ještě poznámka: Další metoda, jak dosáhnout lepšího rozlišení v textovém režimu, je použití ovladače framebuffer (o kterém jsme hovořili v kapitole o X Window). Tato metoda vyžaduje rekonfiguraci jádra (i když některé linuxové distri-buce už vhodné jádro obsahují) a nový záznam (vga=NNN) v souboru /etc/lilo.conf. V texto-vém režimu pak framebuffer funguje i s VESA BIOSy staršími než 2.0.

Poměrně neznámou skutečností je, že čipset NeoMagic NM20xx podporuje textový režim 100x37 (tj. 800x600). Tento textový režim vypadá velmi hezky (ve srovnání s ošklivým režimem 80x25). Máme to vyzkoušeno s notebookem HP OmniBook 800 a postup by měl platit i u jiných notebo-oků s tímto čipsetem.

Jediný problém spočívá v tom, že nastavení je poměrně pracné, a pokud použijete příkazy SVGATextMode nebo restoretextmode špatně, může výsledek na LCD displeji vypadat děsivě. I přes spoustu chybných nastavení se mi sice nepodařilo LCD zničit, ale i tak prohlášení: Násle-dující postup může vést k poškození vašeho hardwaru, varujeme vás! Následující postup zkoušej-te jen na vlastní riziko, nezodpovídáme za žádné případné problémy.

Přehled

Musíte provést *tři* hlavní kroky:

Zapnout v Linuxu bootování v textovém režimu 800x600. Problém je v tom, že dokud neprovedete i následující dva kroky, neuvidíte žádný text.
Automaticky spustit restoretextmode se správnými hodnotami registrů.
Automaticky spustit SVGATextMode.

Podrobněji

Bootování v rozlišení 800x600
Stará jádra (2.2.x) je nutné přeložit s volbou CONFIG_VIDEO_GFX_HACK, která je standardněvypnutá.Starším jádrům předejte parametr vga=770, novějším vga=7. Příklad souboru lilo.conf:

```
image=/boot/bzImage-modif label=22 append="svgatextmode=100x37x8_SVGA" # vysvětlíme později vga=7 read-only
```

Další informace najdete v dokumentaci dodávané ke zdrojových kódům jádra (obvykle v souboru /usr/src/linux/Documentation/fb/vesafb.txt).
Spuštění restoretextmode a SVGATextMode při bootování

Při bootování musíte zajistit spuštění příkazů restoretextmode <název vhodného souboru tex-treg.dat> a SVGATextMode 100x37x8_SVGA.
Příklad souboru textreg.dat (získaného příkazem savetextmode) naleznete v mém archivu. Protože jsem líný, umístil jsem příkazy SVGATextMode a restoretextmode rovnou do souboru /etc/rc.boot/kbd.

Te to důležité

Pokud nepoužijete správný příkaz SVGATextMode ve správném grafickém režimu, budou se zobrazovat hrozné věci. Proto pomocí příkazu append=xxx v lilo.conf předávám jádru pro-měnnou svgatextmode=100x37x8_SVGA (název i hodnota jsou libovolné) v té samé bootovací volbě, kde se nastavuje vga=7. Skript /etc/rc.boot/kbd tuto proměnnou testuje a příkazy res-toretextmode a SVGATextMode spustí tehdy a jen tehdy, je-li správně nastavena.

Postup

Jádro 2.2.x přeložte s volbou CONFIG_VIDEO_GFX_HACK.
Do vhodného inicializačního skriptu bez dalších změn vložte příkaz restoretextmode se správným parametrem.
Nabootujte v normálním textovém režimu (80x25) s proběhnuvším příkazem restoretext-mode: Měli byste vidět, že obrazovka pracuje v režimu 100x37, zobrazuje se však jenom oblast 80x25. Zatím nezadávejte SVGATextMode.

Je rozumné inicializační kód podmínit nějakou proměnnou stejně, jako to mám uděláno já, pak budete mít možnost bootovat v obou režimech. Teď je ta správná chvíle to vyzkoušet a nabootovat s a bez spuštění restoretextmode.
Nabootujte v režimu 100x37 prostřednictvím parametru vga=7 v lilo.conf. V nějaké chví-li byste měli vidět bílé pozadí, znaky se ale nakonec budou vypisovat černá na černé. To je v pořádku, nyní musíte rebootovat naslepo.
Do inicializačního skriptu vložte za příkaz restoretextmode příkaz SVGATextMode 100x37x8_SVGA.
Nabootujte s vga=7.
Teď už by mělo být všechno v pořádku.

Knihy pro linuxové cestovatele

Scott Mueller: Upgrading and Repairing Laptops, 2003

Anotace vydavatele: „Scott Mueller se vydal tam, kam se ještě nevydal žádný autor, přímo za všechny záruční samolepky, skryté šroubky a obavy a vytvořil opravdový uživatelský návod, který by měl mít na stole každý vlastník notebooku. Kniha popisuje, jaké úpravy může provést uživa-tel, jaké je lepší ponechat na servisním pracovišti a jak do různých notebooků přidat periferie. Při-ložené CD obsahuje video toho, co najdete uvnitř notebooku.“

Další informace:

http://repair4laptop.org/,
http://repair4pda.org/,
http://repair4pda.org/,
http://repair4player.org/.

Chris Hurley, Michael Puchol, Russ Rogers, Frank Thornton: WarDriving – Drive, Detect, Defend, A Guide to Wireless Security, 2004

Anotace vydavatele: „WarDriving přivedl špičky bezdrátového průmyslu k napsání skutečně infor-mativní knihy o tom, co to je warDriving, a o nástrojích, které by měl mít ve své výzbroji každý, kdo bezdrátové sítě používá nebo je chce zavádět.

WarDriving je jedinečná kombinace záliby, sociologického průzkumu a zkoumání bezpečnosti. Jízda nebo chůze městem s bezdrátovým přijímačem a mapování chráněných a nechráněných bez-drátových sítí rozpoutala intenzivní debatu mezi právníky, bezpečnostními specialisty a teleko-munikačními odborníky. Tato úplně první kniha o warDrivingu je napsána z pohledu lidí, kteří vytvořili nástroje umožňující warDriving a kteří zjišťují, analyzují a udržují data o zabezpečených i otevřených přístupových bodech ve významné metropolitní oblasti. Dozvíte se od nich také informace o zabezpečení bezdrátové sítě dříve, než ji zneužijí kriminální živly.

Bezdrátové sítě se za posledních několik let staly životním stylem. S rozvojem bezdrátových sítí zároveň roste potřeba jejich zabezpečení. Kniha informuje uživatele i provozovatele bezdrátových sítích o rizicích, která jsou s tímto typem sítí spjatá.“

Další informace:

http://tuxmobil.org/wireless_unix.html,
http://tuxmobil.org/manet_linux.html,
http://tuxmobil.org/wireless_community.html,
http://tuxmobil.org/linux_wireless_access_point.html,
http://tuxmobil.org/linux_wireless_sniffer.html.

Isidor Buchmann: Batteries in a Portable World – A Handbo-ok on Rechargeable Batteries for Non-Engineers, 2001

Anotace vydavatele: „Tato kniha představuje nezbytný zdroj praktických informací o akumuláto-rech. Velmi často vycházejí výkonnostní specifikace baterií a nabíječek z ideálních podmínek. Výrobci testují baterie nové a v chráněném prostředí, bez stresů z každodenního používání. Kniha sleduje každodenní život baterie v rukách běžného uživatele. Díky ní lépe porozumíte silným a slabým stránkám baterií. Dozvíte se, jak prodloužit životnost baterie, seznámíte se s doporučo-vanými způsoby údržby a zjistíte, jak restaurovat slabé baterie v případě, že je to s baterií dané-ho typu možné. Tyto znalosti vám umožní prodloužit životnost baterií, zvýšit spolehlivost pře-nosných zařízení a ušetřit peníze.“

Další informace:

http://tuxmobil.org/energy_laptops.html,
http://tuxmobil.org/mobile_battery.html.

Bob Toxen: Real World Linux Security: Intrusion Detection, Prevention, and Recovery 2nd Ed., 2002

Tato kniha obsahuje kapitolu o bezpečnosti mobilních zařízení. Další informace:
http://tuxmobil.org/mobile_security.html,
http://tuxmobil.org/stolen_laptops.html.

Český překlad knihy Boba Toxena vyšel u nakladatelství Computer Press v roce 2003.

Informace o konkrétních typech notebooků

Některé notebooky jsou v komunitě linuxových uživatelů oblíbenější než jiné. Následující seznam není rozhodně vyčerpávající.

Compaq Concerto Aero

Compaq Concerto Fan's Home Page, *http://www.inetdirect.net/stg/pen/chris/concerto.html*, a Aero FAQ, *http://www.reed.edu/~pwilk/aero/aero.faq.*Poslední verze ovladače pera je k dispozici na *http://www.cs.nmsu.edu/~pfeiffer*, což jsou stránkyJoe Pfeiffera.

Dell

Poštovní konference linux-dell-laptops, *http://tech.groups.yahoo.com/group/linux-dell-laptops/.* Linuxové informace u výrobce, *http://www.dell.com/.*

Lenovo/IBM ThinkPad

ThinkPad Configuration Tool for Linux Thomase Hooda, *http://tpctl.sourceforge.net/.*Konference *Running Linux on IBM ThinkPads*, přihlásit se můžete e-mailem na adresu *linux-think pad-subscribe@topica.com*, další informace naleznete na adrese *http://lists.topica.com/lists/linux-thinkpad/.*Ovladač TrackPointu od Tilla Straumanna, *http://www.slac.stanford.edu/~strauman/pers/tp4utils/.*Výborné informační stránky o Linuxu na noteboocích Thinkpad najdete na *http://www.thinkwiki.org.*

Sony VAIO

Postup instalace přes externí CD mechaniku je popsán výše v kapitole o instalaci. Série Sony VAIO C1 obsahuje některé modely založené na prvním mobilním CPU, CRUSOE. Tento procesor vyrá-bí společnost TransMeta. Na jejich stránkách, *http://www.transmeta.com/*, naleznete informace
o binární kompatibilitě procesoru CRUSOE. Program Sony PCG-C1XS Picturebook Camera Captu-re, *http://samba.org/picturebook/*, snímá obrázky a video na modelech Sony VAIO PCG-C1XS, vyu-žívá vestavěnou CCD kamery a hardwarový JPEG enkodér. Umožňuje zachytávání PPM, JPEG, AVI záznam MJPEG, snímání jednotlivých snímků z MJPEG, nastavení jasu, kontrastu a podobně a možnost sub-samplování 1:4.

K dispozici je také VAIO HOWTO, *http://tldp.org/HOWTO/VAIO+Linux.html* a konference *linux-c1@gnu.org.*
Dokument Sony Vaio C1 FAQ, *http://frijoles.hungry.com/c1-info/faq.html*, je zaměřen převážně na
Windows, obsahuje nicméně užitečné hardwarové informace a odkazy na konference.Sony VAIO SPIC daemon, *http://spicd.raszi.hu/*, je malý hack zprovozňující apmd na noteboocíchSony VAIO. Prostřednictvím jaderného modulu sonypi detekuje stav AC adaptéru a podsvícenídispleje, pomocí cpufreqd řídí rychlost procesoru.

spicctrl, *http://www.alcove-labs.org/en/software/sonypi/*, využívá rozhraní /dev/sonypi.

Toshiba

Toshiba Linux Utilities, *http://www.buzzard.me.uk/toshiba/index.html*, je sada linuxových nástro-jů pro řízení ventilátoru, nastavení hesla správce a obsluhu horkých kláves. K dispozici je také KDE balíček Kliberta.

Poštovní konference linux-on-portege, *http://groups.yahoo.com/subscribe.cgi/linux-on-portege*, a linux-on-toshiba-satellite-40xx, *majordomo@geekstuff.co.uk.*

ČÁST V

Praktické návody HOWTO
Emacs pro začátečníky

V tomto dokumentu seznámíme uživatele Linuxu s editorem Emacs. Předpokladem je alespoň minimální znalost editoru vi nebo podobného. Aktuální verze tohoto dokumentu je běžně dostup-ná na *http://www.wcnet.org/jzawodn/emacs/*.

Komu je příručka určena

Příručka je určena těm uživatelům Linuxu, kteří se chtějí něco dozvědět o Emacsu a vyzkoušet si jej. První návrh vznikl jako stručný výukový program pro setkání linuxových nadšenců v Toledu (Toledo Area Linux User Group), viz *http://www.talug.org/*. Od té doby se dokument díky znač-nému zájmu linuxové komunity poněkud rozrostl. Podrobnosti viz kapitola „Poděkování".

Upřímně řečeno, v tomto dokumentu není nic, co by se týkalo výhradně Linuxu. Je platný pro všechny odrůdy Unixu, a dokonce i pro Emacs provozovaný pod Microsoft Windows. Avšak vzhledem k tomu, že dokument je součástí Dokumentačního projektu systému Linux, považuji za svoji povinnost uvést, že vznikl pro uživatele Linuxu.

Co je Emacs

Emacs je pro každého něco jiného – závisí na kom, koho se zeptáme. Odpovědi mohou být různé:

Textový editor
Poštovní klient
Program na čtení zpráv
Program na zpracování slov
Víra
Integrované vývojové prostředí
Cokoli jiného!

Pro naše účely však mějme za to, že Emacs je textový editor – a to editor velice flexibilní. Později se do tohoto problému ponoříme hlouběji. Emacs napsal Richard Stallman (zakladatel Free Soft-ware Foundation: *http://www.fsf.org/* a projektu GNU, *http://www.gnu.org/*) a udržuje jej dodnes.

Emacs je jedním z nejoblíbenějších a nejsilnějších textových editorů v Linuxu (a v Unixu) – v popu-laritě je na druhém místě za vi. Má množství nejrůznějších nástrojů, snadno se přizpůsobuje růz-ným uživatelským prostředím a je velmi dobře odladěn. Právě z těchto důvodů byl implemento-ván. Aniž bych zabíhal do podrobností, musím říct, že Emacs není jen „obyčejný editor". Je napsa-ný převážně v jazyce Lisp, jeho jádro je plnohodnotným interpretem Lispu napsaným v jazyce C. V editoru samotném jsou v jazyce C napsány pouze nejzákladnější části na nejnižší úrovni, jinak je většina programu napsána v Lispu. Emacs má v sobě vlastně „vestavěný" kompletní programovací jazyk, jehož prostřednictvím jej můžeme přizpůsobovat, rozšiřovat a měnit jeho chování.
Emacs je současně i jedním z nejstarších editorů. Skutečnost, že jej už dvacet let (?) používají tisí-ce programátorů, má za následek, že k němu existuje mnoho doplňujících balíků. Díky nim může Emacs provádět věci, o nichž se Stallmanovi pravděpodobně ani nesnilo, když na Emacsu začal pracovat. Ostatní podrobnosti v dalším textu.

Na Internetu existuje mnoho stránek a dokumentů, v nichž nalezneme lepší popis Emacsu, jeho historie a všeho, co se k němu vztahuje. Nebudeme si je uvádět na tomto místě, je jim vyhrazena kapitola „Jiné zdroje".

Původ a verze

Je nutno zdůraznit, že ve skutečnosti existují dva různé editory Emacs: GNU Emacs a XEmacs. Oba mají stejný původ a mnoho společných vlastností. Zde se zaměříme na GNU Emacs (konkrétně verze 20.3), avšak většina uvedených údajů platí i pro

XEmacs a dřívější verze GNU Emacs. Pro-sím laskavé čtenáře, aby vzali na vědomí, že v tomto dokumentu se budu většinou odkazovat pouze na „Emacs“. Většina distribucí má v balíčcích obě verze Emacsu.

Kde jej najdeme

Emacs najdeme snadno. V nejčastěji používaných distribucích jako Debian, RedHat, Slackware apod. bude Emacs nejspíše volitelným balíkem, který si můžeme nainstalovat z distribučního média. Pokud ne, můžeme si stáhnout zdrojový kód Emacsu a sami si jej přeložit. Kde je přesně uložen, najdeme na *http://www.gnu.org/software/emacs/emacs.html*.

Poznámky k návodu

V tomto návodu jsme z praktických důvodů ponechali typografické konvence pro názvy pro-měnných a klávesové zkratky stejné, jako používá Emacs, protože předpokládáme, že odlišné zna-čení by ztížilo orientaci v začátcích. Nezapomeňte tedy, že klávesová zkratka C-x znamená to samé, co jinde v knize Ctrl+x.

Provozování editoru Emacs

Spuštění & ukončení editoru Emacs

Nový uživatel si asi nejdříve spustí Emacs jen tak, aby si ho prohlédl a něco si s ním zkusil. Může se ale stát, že si jej spustíme a nebudeme vědět, jak jej ukončit. Pracujeme-li s tímto editorem popr-vé, nejdříve si jej vyzkoušíme. Když se ohlásí shellový prompt, zadáme emacs a stiskneme klá-vesu Enter. Tím by se měl Emacs spustit. Pokud se nespustí, tak buď není nainstalován nebo není v cestě.

Dále se musíme naučit Emacs ukončovat. Emacs opustíme pomocí kláves C-x C-c. Zápis C-x zna-mená přidržet klávesu Ctrl a stisknout x. V tomto případě musíme přidržet Ctrl a stisknout c, aby-chom úlohu ukončili.

Klávesy používané v Emacsu se nám možná ze začátku budou zdát nešikovně zvolené, nebo dokonce nepohodlné, zejména jsme-li zvyklí na editor vi. Emacs na rozdíl od vi nemá zvláštní režim pro editaci textu a pro zadávání příkazů.

Rekapitulace: Emacs se spouští pomocí příkazu emacs a ukončuje pomocí kláves C-x C-c.

Co uvidíme

Emacs po spuštění zabere celé okno X Window (resp. obrazovku, pracujeme-li s konzolou). Na horním okraji se objeví nabídka, uprostřed obrazovky nějaký text a na spodním okraji obrazovky několik řádků.
V ASCII znacích si můžeme znázornit obrazovku asi takto:

```
+------------------------------------------------------------------------+ |Buffers Files Tools Edit Search Mule Help | | | |Welcome to GNU Emacs, one component of a Linux-based GNU system.  |
|                                                                        |
|                                                                        |
|                                                                        |
| ...                                                                    |
|                                                                        |
|---1:---F1          *scratch*                   (Lisp Interaction)--L1--All----------|
|For information about the GNU Project and its goals, type C-h C-p.  |
+------------------------------------------------------------------------+
```

Poznámka

Emacs obvykle zaplní celou obrazovku, resp. celé okno. Zápis shora je z důvodu úspory místa zkrácený. Při prvním spuštění také uvidíme uvítací zprávu, která je vynechaná a nahrazená třemi tečkami. Obsahuje jednoduchou identifikaci spuštěné verze Emacsu, odkaz na nápovědu apod.

Panel s nápovědou

Horní řádek rozhraní Emacsu obsahuje nabídku. Provozujeme-li systém X Window, bude mít tvar obvyklé roletové nabídky, z níž vybíráme myší. Také lze používat klávesové zkratky (zde se o nich nezmiňujeme).

Stavový řádek a minibuffer

Hořejší ze dvou posledních řádků rozhraní Emacsu je v podstatě stavový řádek. Obsahuje infor
mace o pracovním bufferu, režimu a mnoho dalších údajů. Prozatím jej stačí vzít na vědomí. Spodní řádek se nazývá *minibuffer*.

Od hlavního bufferu jej dělí právě popsaný stavový řádek. Minibuffer si můžeme představit jako „příkazový řádek“. Zobrazují se v něm právě zadávané pří-kazy a stavová hlášení jako odpovědi na tyto příkazy.

Brzy zjistíme, že na to, co jsem nazval stavovým řádkem, se příslušná dokumentace Emacsu odkazuje jako na režimový řádek. Jsou tam totiž zobrazeny informace o stávajícím režimu (stávajících režimech) a můžeme jej využít i na zobrazení data a času, čísla řádku, velikosti souboru a téměř čehokoli, co bychom tam chtěli mít.

Něco o terminologii

V této části se zmíníme o základní terminologii Emacsu, s níž se budeme setkávat.

Buffery a soubory

Na rozdíl od některých jiných editorů nejsou soubory, s nimiž pracuje Emacs, otevřené až do ukončení práce. Emacs si je načte do *bufferu* v paměti a pracuje s nimi zde, přičemž na disku se nic nemění. Změna nastane až při uložení bufferu na disk. Tento způsob práce se soubory má svoje výhody i nevýhody a musíme s touto skutečností počítat.

V důsledku toho se budeme v dokumentaci Emacsu, v popisech režimů a v souvisejících balících apod. setkávat s výrazem „buffer“. Musíme si uvědomit, že „buffer“ znamená „kopie souboru v paměti“. V této souvislosti je však také nutno poznamenat, že toto není jediný buffer, s nímž Emacs pracuje. Buffery si vytváří také pro zpracování příkazů, výsledků, různých seznamů apod.

Ukazatel a oblast

V emacsové hantýrce se také často setkáme s výrazem *ukazatel* (*point*). Obecně vzato jde o kur-zor. Pro začátek nejspíš nebude nutné rozlišovat mezi ukazatelem a kurzorem, avšak puntičkářům je nutno říci, že kurzor je vizuální reprezentace ukazatele. Kurzor je vždy na pozici některého znaku, zatímco ukazatel zásadně ukazuje *mezi znaky* v bufferu. Je-li tedy kurzor ve slově „the“ na pozici písmene „h“, ukazatel ukazuje mezi písmena „t“ a „h“.

Podobně jako je tomu u mnoha moderních editorů, Emacs může provádět operace (zalomení, pře-formátování, zrušení, kopírování, přidání, ...) s oblastmi v běžném bufferu. Pomocí myši nebo klá-ves můžeme určitý blok textu zvýraznit („označit“) a pak provádět operace pouze s ním. V Emac-su se blok textu nazývá *oblast* (*region*).

Okna

Uživatelům grafických rozhraní (GUI) teď do celé věci nejspíš vneseme trochu zmatku. Musíme si však uvědomit, že Emacs vznikl mnohem dřív, než přišli do módy správci oken a grafická rozhraní.

Okno v Emacsu je oblast obrazovky, v níž je zobrazen buffer. Při prvním spuštění Emacsu bude-me mít na obrazovce jen jedno okno. Některé funkce (nápověda, dokumentace) si na obrazovce často otevírají svá vlastní [dočasná] okna.

Emacsové okno nemá nic společného s okny systému X Window ve smyslu grafického rozhraní. V systému X Window si můžeme průběžně otvírat další okna a třeba v nich porovnávat soubory. V hantýrce Emacsu nazýváme tato okna *rámce* (*frames*). Další se dozvíme později.

Rámce

Rámec je v Emacsu samostatné okno X Window, v němž je zobrazen buffer. Obě okna však jsou součástí jednoho sezení Emacsu. Trochu se podobají tomu, co se stane, když v prohlížeči Nets-cape Navigator stiskneme Alt+N.

Klávesnice

V této části si popíšeme základy práce s klávesnicí v editoru Emacs. Jako v každém výkonném editoru můžeme vše udělat stisknutím několika kláves. Uživatelé editoru vi by mně pravděpodobně potvrdili, že na pohyb po řádcích nahoru, dolů,vpřed a vzad pomocí kláves k, j, l, h si museli chvilku zvykat. Ve skutečnosti to může trvat i něko-lik hodin, nebo dokonce i týdnů, než si člověk ve vi osvojí dokonalou navigaci po souboru pomo-cí různých kombinací kláves.

Totéž platí i pro Emacs. Musíme se naučit používat různé klávesy a příkazy. Podobně jako ve vi jich budeme pro většinu práce potřebovat minimum a teprve časem budeme přicházet na to, jak bychom některé věci mohli provést rychleji.

Příkazové klávesy (Meta, Esc, Control a Alt)

Jak jsme se už zmínili, v editoru Emacs se často používají klávesové zkratky. Avšak vzhledem k tomu, že Emacs nepracuje v různých režimech jako například vi, tak když budeme chtít posu-nout kurzor nebo zadat příkaz, nebudeme v „příkazovém režimu“ nebo „edičním režimu“. Pouze stiskneme správnou kombinaci kláves a Emacs (možná) udělá to, co po něm chceme.

Nejčastěji používané klávesy jsou v dokumentaci obvykle zkráceny jako C (jako Control, resp. Ctrl) nebo M (jako Meta). Zatímco většina současných klávesnic u osobních počítačů má jednu nebo několik kláves označených Ctrl, jen málo jich má klávesu Meta. Musíme si ji v duchu nahradit klávesami Esc nebo Alt. Ve standardních konfiguracích provádějí Esc i Alt v podstatě totéž. Když tedy uvidíme v dokumentaci

Emacsu zápis C-x f, znamená „stiskni Ctrl+x a pak f“. Zápis M--x shell pak znamená „stiskni Alt-x a zadej slovo shell“.

Velice užitečným příkazem pro začátečníky je M-x apropos, resp. C-h a. Tímto příkazem najdeme v on-line dokumentaci všechny funkce a regulární výrazy, které zadáme. Je to velmi vhodný způ-sob, jak najít všechny příkazy, které se vztahují k rámcům. Zadáme prostě C-h a a pak frame.

Jak se pohybujeme po bufferu

Teď už tedy víme, co všechny ty podivné zkratky znamenají, takže si můžeme uvést seznam nejčastěji používaných kláves pro pohyb v bufferu:

Klávesy	Akce
C-p	O jeden řádek nahoru
C-n	O jeden řádek dolů
C-f	O jeden znak vpřed
C-b	O jeden znak vzad
C-a	Začátek řádku
C-e	Konec řádku
C-v	O jednu stránku nahoru
M-v	O jednu stránku dolů
M-f	O jedno slovo vpřed
M-b	O jedno slovo vzad
M-<	Začátek bufferu
M->	Konec bufferu
C-g	Zrušení stávající operace

Klávesy pro pohyb kurzoru (šipky) pracují, jak bychom mohli očekávat. To ovšem nemusí platit pro klávesu Backspace, ale to už je jiná povídačka.

Základní příkazy

Tak, a když už se umíme pohybovat v bufferu, měli bychom si něco říci o otevírání a ukládání souborů a o hledání. Uvedeme několik základních příkazů.
Ještě předtím se však musíme zmínit o tom, jak pracují. Všechny „příkazové klávesy“ v Emacsu (typu M-x něco, resp. C-něco) jsou ve skutečnosti jen zkratkami funkcí, jež jsou součástí Emacs. Všechny můžeme vyvolat tak, že zadáme M-x jméno-funkce a stiskneme klávesu Enter anebo můžeme použít klávesovou zkratku dané funkce (pokud existuje).

Například funkce pro uložení bufferu na disk se jmenuje save-buffer. Implicitní klávesová zkrat-ka je C-x C-s. Stávající buffer tedy můžeme uložit pomocí klávesové zkratky anebo tak, že zadá-me příkaz M-x save-buffer. Výsledek bude stejný.
Všechny často užívané funkce mají implicitní klávesové zkratky. Zde jsou některé z nich.

Klávesy	Funkce	Popis
C-x C-s	save-buffer	Ulož stávající buffer na disk
C-x u	undo	Zruš poslední operaci
C-x C-f	find-file	Otevři soubor na disku
C-s	isearch-forward	Hledej řetězec vpřed
C-r	isearch-backward	Hledej řetězec vzad
	replace-string	Hledej a nahra řetězec
	replace-regexp	Hledej a nahra pomocí regexp
C-h t	help-with-tutorial	Použij interaktivní návod
C-h f	describe-function	Zobraz nápovědu k funkci
C-h v	describe-variable	Zobraz nápovědu k proměnné
C-h x	describe-key	Vypiš, co dělá klávesová sekvence
C-h a	apropos	Najdi nápovědu pro řetězec/reg. výraz

C-h F	view-emacs-FAQ	Vypiš často kladené otázky (FAQ) v Emacs
C-h i	info	Čti dokumentaci Emacs
C-x r m	bookmark-set	Nastav záložku. Používá se při hledání
C-x r b	bookmark-jump	Jdi na záložku

Když si funkce vyzkoušíme, zjistíme, že řada z nich bude požadovat nějaký vstup, což provedou v minibufferu. Používá se podobně jako příkazy v editoru vi nebo většina příkazů, které používáme v našem oblíbeném unixovém shellu.

Emacs má doslova stovky nejrůznějších vestavěných příkazů. Seznam shora je jenom malou ukáz-kou toho, co pravidelně používám. Obsáhlejší seznam funkcí a úplnější dokumentaci funkcí uve-dených výše naleznete v on-line nápovědě.

Doplňování příkazů

Podobně jako je tomu u oblíbených shellů (bash, csh, tcsh, ...), nabízí Emacs doplňování příkazů pomocí klávesy Tab. Ve skutečnosti bylo doplňování příkazů v bashi vytvořeno podle vzoru Emacs, takže uživatelé shellu bash budou v tomto ohledu v Emacsu doma.

Můžeme si například vyzkoušet příkaz pro hledání M-x, poté stiskneme Tab. Emacs přidá k pří-kazu pomlčku, čímž sdělí, že existuje několik možných doplnění, avšak všechna mají jako následu-jící znak pomlčku. Když stiskneme znovu klávesu Tab, vypíše se seznam možných shod, z nichž si jednu vybereme. Všimněme si, že vše se děje v *novém okně*. Zadávání bude dočasně probíhat ve dvou oknech: Jedno bude obsahovat editovaný buffer a druhé seznam možných doplnění „search-“. Proces výběru můžeme ukončit pomocí C-g, nové okno se zavře.

Návod, nápověda & Info

V editoru Emacs je k dispozici on-line návod, který seznamuje uživatele se základními možnostmi a funkcemi editování, jež by měl každý znát. Také poskytuje vysvětlení, jak používat nápovědu. Chceme-li se Emacs opravdu naučit, měli bychom nějaký čas strávit prohlížením tohoto návodu. Jak je uvedeno v tabulce shora, návod otevřeme příkazem C-h t. Je určen začátečníkům a sám nabízí postup při prohlížení látky.
Provozujeme-li Emacs v systému X Window, vidíme, že menu zcela vpravo na panelu je označe-no Help. Všimněte si, že při hledání v nápovědě mají některé položky klávesové zkratky, které se zobrazují na pravé straně.

Chceme-li se seznámit s rozsahem dokumentace Emacsu, můžeme zadat příkaz M-x info nebo klávesovou zkratku C-h i, jimiž spustíme Info, což je prohlížeč dokumentace v Emacsu.

Režimy Emacsu

Režimy Emacsu se rozumí různá chování a vlastnosti, jež můžeme zapnout nebo vypnout (anebo pochopitelně přizpůsobit) z důvodu používání v různých prostředích. Tím, že nastavíme různé režimy, můžeme jedním editorem psát dokumentaci, programovat v různých jazycích (C, C++, Perl, Python, Java a mnoho dalších), vytvořit domovskou stránku, posílat elektronickou poštu, číst zprávy na Usenetu, organizovat pracovní schůzky, a dokonce i hrát hry.

Režimy Emacsu jsou knihovny kódu v Lispu, které rozšiřují, modifikují nebo určitým způsobem obohacují Emacs.

Hlavní a vedlejší režimy

V zásadě existují dva druhy režimů: hlavní a vedlejší. Vysvětlit rozdíly mezi nimi není úplně trivi

ální, zvláště tomu, kdo s nimi ještě nepracoval, avšak pokusíme se o to.V daném okamžiku může být aktivní pouze jediný hlavní režim, zatímco vedlejších režimů můžebýt současně aktivních několik. Hlavní režimy se vztahují k určitému jazyku nebo k určité úloze,zatímco vedlejší režimy jsou kratší a méně vyhraněné utility, jež se mohou využívat v nejrůzněj-ších úlohách.

Zní to poněkud abstraktně, zkusme si tedy uvést příklad. Existuje režim, který používám docelačasto, když píši prostý text. Jmenuje se text-mode a je určen pro psaní volného textu, jako napří-klad soubor README. Umí rozpoznávat slova a odstavce a obecně dělá to, co očekávám, že budedělat, když budu používat běžné navigační klávesy.

Když píši text, který má být čtivý, obvykle chci, aby vypadal lépe. Měl by být nějak rozumně zalo-mený atd. Zalamování slov aktivuji pomocí vedlejšího režimu auto-fill. Tento režim se snažívhodně upravovat text, i když píši pořád dál a narazím na konec řádku. Skutečnost, že je vedlej-ším režimem, znamená, že může pracovat v několika různých hlavních režimech. Shora

zmíněná„vhodná úprava“ ovšem vypadá jinak například v režimu text-mode a jinak v režimu java-mode.Nechci, aby kód v jazyce Java byl zalamován po slovech jako v normálním textu. Chci však, abytakto byl zalomen v komentářích! Režim auto-fill je natolik chytrý, že umí zalamovat text tímtozpůsobem.
Autoři různých režimů v Emacsu si dali hodně záležet na tom, aby vedlejší režimy byly skutečně

vedlejší.Když se podíváme na obrazovku Emacsu uvedenou na straně 629, vidíme, že na režimovém řádkujsou uvedeny režimy, v nichž se Emacs nalézá. V tomto případě je to režim „Lisp Interaction“, cožje implicitní režim, který se skutečně hodí jen k zápisu kódu v Lispu. (Avšak vzhledem k tomu,že Emacs je převážně napsaný v Lispu, proč ne?)

Programovací režimy

Editor Emacs hlavně a především vymyslel programátor pro programátory. Má vysoce kvalitní reži-my prakticky pro všechny oblíbené programovací jazyky, které si umíme představit (a také pro několik méně oblíbených). Zde se zmíníme jen o některých. Většina programátorských režimů má některé společné rysy. Obvykle dělají něco (nebo všechno) z následujícího seznamu:

Zvýrazňují syntaxi v daném jazyce.
Automaticky zalamují a formátují kód v daném jazyce.
Mají kontextovou nápovědu pro daný jazyk.
Automaticky spolupracují s ladicím programem.
Na nabídkovém panelu mají speciální menu pro daný jazyk.

Dále existuje několik pomocných, jazykově nespecifických režimů, jež jsou společné pro různé jazyky. Jde například o rozhraní na software pro správu verzí, automatické přidávání komentářů, vytváření souborů Makefile, aktualizace souborů s logy apod.

Když vezmeme všechny tyto režimy a zvážíme, jak je kód Emacsu prověřený a stabilní, můžeme jej klidně srovnat s komerčně šířenými IDE (*Integrated Development Environments*) pro takové jazyky jako C++ nebo Java. A pochopitelně je zdarma.

C/C++/Java

Vzhledem k tomu, že syntaxe jazyků C, C++ a Java jsou podobné, existuje pro ně v Emacsu pouze jeden režim (podobně jako pro Objective-C a IDL). Jde o velmi osvědčený a komplexní balík, který je součástí distribuce Emacsu. Tento režim se nazývá cc-mode, resp. CC Mode.

Další podrobnosti a nové verze naleznete na *http://www.python.org/emacs/*.

Perl

Pro editování kódu perl existují v Emacsu dva režimy: První se nazývá (jak bychom asi předpo-kládali) perl-mode, druhý pak cperl-mode. Důvody, proč existují dva režimy, příliš dobře neznám (v dokumentaci o tom nic není), ale zdá se, že původním režimem pro editaci kódu Perl v Emacsu byl perl-mode. Zdá se, že má méně funkcí než cperl-mode a že není schopen rozpo-znávat některé složitější konstrukce jazyka Perl.

Já osobně používám a doporučuji režim cperl-mode, který je velice aktivně udržován a má prak-ticky veškeré náležitosti, které jsem kdy potřeboval. Nejnovější verze lze nalézt na ftp://ftp.math.ohio-state.edu/pub/users/ilya/emacs.

Neberte mě však za slovo zcela bezvýhradně. Vyzkoušejte je oba a vyberte si ten, který vám bude lépe vyhovovat.

Python

Python (další velmi oblíbený jazyk na psaní skriptů) má v Emacsu také svůj režim. Pokud vím, není distribuován s editorem GNU Emacs, nýbrž jen s XEmacsem. Pracuje v obou editorech. Režim python-mode nalezneme na oficiálních stránkách Pythonu.

http://www.python.org/emacs/python-mode/.

Jiné

Programátoři naleznou v Emacsu mnoho dalších zajímavých režimů. Jsou určeny například pro:

shellové skripty (Bash, sh, ksh, csh, …),
Awk, Sed, Tcl, ...,
soubory Makefiles,
změny v logu,
dokumentaci,
ladění.

A mnoho dalších. V poslední části tohoto dokumentu jsou uvedeny informace o dalších režimech a doplňcích.

Tvorba textů

Nejrůznější režimy Emacsu nejsou omezeny pouze na psaní kódu. Kdo píše dokumentaci (jakéhokoli druhu), může si vybrat z bohaté nabídky.

Kontrola pravopisu (režim ispell)

Autoři mnoha typů dokumentů potřebují čas od času zkontrolovat gramatiku. Máme-li nainstalo-vaný GNU ispell, pro kontrolu gramatiky běžného bufferu stačí zadat příkaz M-x. Když ispell nalezne slova, která nezná, nabídne seznam možných náhrad, z nichž můžeme vybrat jednu nebo žádnou. Je funkčně ekvivalentní s kontrolami gramatik v mnoha oblíbených, komerčně prodáva-ných balících.

HTML (režim html-helper)

Píšeme-li občas (nebo dokonce často) soubory HTML, není od věci vyzkoušet režim html-hel-per-mode. Nalezneme jej na *http://www.santafe.edu/~nelson/tools/*, kde je jak dokumentace, tak i vše ostatní.

Jak už sám název napovídá, tento režim je určen těm, kteří prozatím píší soubory HTML „ručně“ postaru.

TeX (tex-mode)

Píšeme-li dokumenty v Texu, můžeme pomocí tohoto režimu svoji práci trochu vylepšit. Díky reži-mu tex-mode v textu přibudou barvy a zvýrazní se obrácená lomítka, složené závorky a některé další znaky.

I když TeX už tolik nepoužívám, mohu potvrdit, že v tomto režimu jsou zdrojové soubory o něco čitelnější.

SGML (sgml-mode)

Dokument, který máte před sebou, byl napsán v SGML (a poté pravděpodobně převeden do tvaru, v němž jej právě čtete). Režim sgml-mode nabízí vše, co potřebujeme pro tvorbu dokumentů: kon-trolu platnosti, dopředný a zpětný posuv přes tagy, zvýrazňování a mnohé další. Tento režim je standardní součástí Emacsu.

Další režimy

Existuje pochopitelně mnoho dalších šikovných režimů, které nám mohou usnadnit život. Některé oblíbené si uvedeme:

Správa verzí (režim vc)

Režim vc je kompatibilní s většinou oblíbených procedur pro správu verzí (RCS, SCCS, CVS), které se používají k vytváření nových a rušení starých souborů, kontrolám verzí apod. Je standardní sou-částí Emacsu včetně dokumentace.

Režim Shell

Proč bychom se měli přepínat do dalšího X Window okna nebo na virtuální konzolu kvůli něko
lika shellovým příkazům? Abychom si ušetřili spoustu starostí.Příkazem M-x shell spustíme shell v rámci bufferu Emacsu. Můžeme v něm provádět mnohemvíce věcí než v normálním shellu (kromě spouštění programů, které využívají celou obrazovku,jako např. vi nebo pine), neboť Emacs spolupracuje se shellem „za scénou“.

Je rovněž standardní součástí Emacsu včetně dokumentace.

Telnet a FTP

Proč bychom se měli přepínat do dalšího X okna nebo na virtuální konzolu kvůli spouštění pro
gramu telnet nebo FTP? Abychom si ušetřili spoustu starostí. (Co vám to připomíná?) Podobně jako je tomu u shellu, můžeme v rámci editoru Emacs provozovat telnet a ftp. Zadáme příkaz M-x telnet nebo M-x ftp a můžeme si to vyzkoušet. Podrobnosti viz dokumentace.

Man

Proč bychom se měli přepínat do dalšího X Window okna nebo na virtuální konzolu kvůli čtení
manuálových stránek? Abychom si ušetřili spoustu starostí. (Slibuji, že už s tím dám pokoj.) Podobně jako je tomu u shellu, můžeme v rámci editoru Emacs číst manuálové stránky. Zadáme příkaz M-x telnet nebo M-x man a můžeme si to vyzkoušet. Podrobnosti viz dokumentace.

Ange-FTP

Ocitujeme dokumentaci ange-ftp:Tento balík slouží k co možná nejjednoduššímu a nejprůhlednějšímu zpřístupnění souborů a adre-sářů FTP z GNU Emacsu. Množina rutin z tohoto balíků spolupracuje přímo s FTP, abychom mohlisoubory zpracovávat, jak jsme zvyklí.To znamená, že můžeme se soubory na vzdáleném počítači nakládat jako s lokálními. Když napří-
klad chceme editovat soubor na jiném počítači, pouze jej v Emacsu otevřeme (odlišná je pouzesyntaxe cesty) a on už se postará o všechny náležitosti spojené s přihlašováním a přenosem sou-boru. Když pak soubor pomocí příkazů C-x C-s uložíme, ange-ftp toto uložení zprostředkujea soubor zapíše na vzdálený počítač.

Odlišnost syntaxe cesty spočívá v tom, že soubor jménem mujsoubor v adresáři users na počí-tači my.host.org otevřeme (C-x f)

jako soubor:

/user@my.host.org:~user/myfile

Tento režim je rovněž standardní součástí distribuce Emacsu včetně dokumentace. Za příklad shora děkuji Etiennu Grossmannovi (*etienne@anonimo.isr.ist.utl.pt*).

Individuální přizpůsobování (modifikace) Emacsu

Prakticky veškerá individuální přizpůsobení editoru Emacs lze provést v kódu jazyka Lisp. Může-me buď měnit proměnné, jež ovlivňují činnost Emacsu, nebo můžeme k Emacsu přidávat nové funkce (anebo přepisovat funkce stávající, tj. nahrazovat je vlastními).

Dočasná přizpůsobení

Při experimentování s modifikacemi Emacsu začneme nejspíš s dočasnými změnami. Pokud pro-vedeme něco strašného, Emacs jen ukončíme příkazem C-x C-c a spustíme jej znovu. Jestliže pro-vedeme změny, které mají být trvalé, přidáme je k vlastnímu souboru ~/.emacs tak, aby se změny projevily při každém dalším spuštění. O tom si řekneme více v další části.

Dosazování do proměnných

Nejsnazší modifikaci provedeme změnou hodnoty proměnné v Emacsu. Kód vypadá takto:

```
(setq variable-name new-value)
```

kde variable-name je jméno proměnné a new-value je hodnota, již chceme dát proměnné. (V řeči jazyka Lisp proměnnou svážeme s hodnotou.) Funkce setq v programu lisp je analogická dosazovacímu operátoru (obvykle =) v jiných programovacích jazycích.

> Poznámka
>
> Pro zjednodušení zde přeskakuji řadu podrobností. Může se stát, že někdy použiji lispo vou funkci set, nebo dokonce setq-default. Koho to opravdu zajímá, může se podívat do emacsového manuálu jazyka Lisp.

Podívejme se nyní na řádek z mého souboru .emacs

```
(setq-default transient-mark-mode t)
```

Proměnnou transient-mark-mode se ovládá zvýraznění označené oblasti. Ve většině grafických aplikací vybereme oblast textu tak, že klepneme a potáhneme myší a oblast se zvýrazní například inverzí nebo nějakou barvou. Když v Emacsu nastavíme proměnnou transient-mark-mode na neprázdnou hodnotu, dělá totéž.

Hodnota WHAT?

Malá odbočka. Většina programovacích jazyků má nějaké vyjádření hodnot true/false (pravda/nepravda). V jazyce C/C++ považujeme hodnotu za pravdivou (true), je-li nenulová. V jazyce Perl musí být neprázdná a nenulová. V Lispu existuje něco podobného, pouze názvy a symboly jsou jiné.

True se obvykle zapisuje jako t a false (resp. null) jako nil. A jako v ostatních jazycích se hod-nota různá od nil považuje za pravdivou. Úplný popis funkce proměnné transient-mark-mode nalezneme v on-line nápovědě. Stačí zadatC-h v nebo M-x describe-variable a pak transient-mark-mode. Je-li někdo tak líný jako já,jméno proměnné si doplní pomocí klávesy Tab. Zadáme pouze část jména proměnné a stiskne-me klávesu Tab. Je-li ze zadané části možné proměnnou jednoznačně identifikovat, zbytek sedoplní.

Jinou často používanou proměnnou je fill-column. Určujeme jí šířku obrazovky pro zalamová-ní (její hodnotou se řídí i režim auto-fill-mode). Kdybychom tuto proměnnou nastavili nanesmyslnou hodnotu, například:

```
(setq fill-column 20)
```

nestane se vůbec nic. Musíme Emacsu říct, aby vyhodnotil zadaný výraz. Provedeme to tak, že umístíme ukazatel (kurzor) na konec tohoto výrazu a zadáme C-x C-e, což vyvolá funkci eval-last-sexp v místě, kam ukazuje kurzor. Jakmile to provedeme, v minibufferu na spodním okra-ji obrazovky se vypíše 20 (resp. jiná zadaná hodnota) jako výsledek vyhodnocení výrazu.

Funkci ověříme tak, že zadáme pár vět. Máme-li aktivovaný režim auto-fill-mode (což asi nemá-me), uvidíme, že se text zalamuje na 20. sloupci. Jinak, po napsání kousku textu, zadáme pří-kaz M-q, který vyvolá funkci fill-paragraph, jež zalomí text.

Spojování souborů a režimů

Emacs můžeme zkonfigurovat tak, že při otevření soubor určitého typu něco provede automatic-ky (třeba jako když v grafickém prostředí klepneme na ikonu, která patří nějakému souboru, a spustí se určitá aplikace). Mohu například chtít, aby se Emacs při

otevření souboru s příponou .txt automaticky přepojil do režimu text-mode. To už se skutečně děje. Nechť tedy Emacs pře-jde do režimu text-mode, když otevřeme soubor „README“.

```
(setq auto-mode-alist (cons ‘(“README” . text-mode) auto-mode-alist))
```

Opravdu? Aniž bychom příliš zabíhali do programování v Lispu, o němž nepotřebujeme nic vědět (avšakneuškodilo by nám), řekněme si jen, že proměnná auto-mode-alist obsahuje seznam dvojic.
Každá dvojice obsahuje regulární výraz a jméno režimu. Je-li otevřený soubor shodný s regulár-ním výrazem (v tomto případě s řetězcem README), Emacs přejde do uvedeného režimu. Legrační syntaxe shora ve skutečnosti znamená, že jsme do seznamu režimů přidali další dvojici.
Nestačí jen dosadit do auto-mode-alist, aniž bychom se přesvědčili, že jsme zrušili původní hodnotu.A kdybych chtěl, aby se Emacs pokaždé při otevření souboru s příponou .html nebo .htm auto-maticky přepínal do režimu html-helper-mode, přidám do svého souboru .emacs:

```
(setq auto-mode-alist (cons ‘(“\\.html$” . html-helper-mode) auto-mode-alist)) (setq auto-mode-alist (cons ‘(“\\.htm$” . html-helper-mode) auto-mode-alist))
```

Možností je skutečně nekonečně mnoho.

Soubor .emacs

Když už jsme věnovali tolik času individuálnímu přizpůsobování editoru Emacs a pochopili jsme jeho hlavní myšlenku, mohli bychom si říci, jak provedeme trvalé změny (anebo platné po dobu, dokud se nerozhodneme jinak). Používáme-li Emacs denně, určitě si všimneme, že se soubor .emacs časem neustále zvětšuje. To je dobře, neboť to znamená, že jsme se naučili přizpůsobo-vat Emacs svým potřebám. A je hanba, že jiný software takovou možnost nenabízí.

Možná jste na to ještě nepřišli, ale pokaždé, když spustíme Emacs, systém nejdříve hledá soubor jménem .emacs v domovském adresáři. Náš soubor .emacs je tam, kam normálně ukládáme kód v jazyce Lisp, který se má spouštět automaticky, a obsahuje individuální přizpůsobení, která jsme zde probírali.

Soubor .emacs obsahuje mimo jiné:

```
(setq inhibit-startup-message t)
```

Proměnná inhibit-startup-message říká, zda má Emacs při spuštění vypsat uvítací zprávu. Když už mě tato zpráva časem omrzí (protože už dávno vím, jak se hledá nápověda apod.), začnu se zajímat o to, jak ji zrušit.

Zkuste si z cvičných důvodů vytvořit vlastní soubor .emacs a přidat do něj řádek, kterým zrušíte vypisování uvítacího řádku. Potom Emacs ukončete a spusťte znovu. Uvítací zpráva by se neměla objevit.
Občas, když čteme o režimech (nebo balících) Emacsu, narazíme v dokumentaci na kód, který můžeme přidat do souboru .emacs a upravit tak určitým způsobem chování editoru. Mnoho takových úprav nalezneme v části FAQ (často kladené otázky, C-h F) přímo v GNU Emacs.

Balík individuálních přizpůsobení

Jak rostla obliba Emacsu a pokračoval jeho vývoj, někdo si řekl, že „musí přece existovat lepší způsob, jak by si mohl začátečník přizpůsobit Emacs“. A balík funkcí *customize* byl na světě. Tento balík nabízí intuitivnější metodu, jak přizpůsobit součásti Emacsu. Vyzkoušíme jej tak, že buď navštívíme část nabídky Customize v nápovědě nebo zadáme příkaz M-x.

Jednotlivá přizpůsobení jsou seskupena do logických skupin jako „Editing“, „Programming“, „Files“ atd. Některé skupiny obsahují další podskupiny.Pokud provádíme změny pomocí tohoto rozhraní, Emacs je uloží do souboru .emacs. To je šikov-né, neboť si jednotlivé změny můžeme prohlédnout (a případně je změnit).

Sám toto rozhraní nepoužívám, takže o něm nedokážu říct nic víc.

Zobrazování v okně X Window

Jako každá správně vytvořená aplikace, i Emacs respektuje prostředky systému X Window. To znamená, že si sami řídíme nastavení barev, geometrii a další systémové záležitosti jako v xterm, nxterm apod.

Zde je příslušná část mého souboru ~/.Xdefaults file:

```
emacs*Background: DarkSlateGray emacs*Foreground: Wheat
emacs*pointerColor: Orchid emacs*cursorColor: Orchid
emacs*bitmapIcon: on emacs*font: fixed emacs.geometry: 80x25
```

Podrobnosti o dalších prostředcích systému X Window viz manuál.Chris Gray (*cgray4@po-box.mcgill.ca*) k tomu také poznamenává: Nezdá se, že by se v systému Debian používal soubor ~/.Xdefaults. Nicméně, uživatelé syste-

mu Debian uloží potřebné do /etc/X11/Xresources/emacs a mají stejně pěkné barvy, jako když používali systém RedHat.

Oblíbené balíky

Kromě různých režimů v Emacsu také existuje celá řada doplňkových *balíků*. Nazývám je balíky proto, že jsou něčím víc než jenom novými režimy. Často obsahují vlastní nástroje anebo jsou tak rozsáhlé, že nazývat je jenom režimy by nebylo spravedlivé. A ještě v jiných případech jsou soft-warem, který rozšiřuje nebo integruje jiné emacsové režimy a balíky. Rozdíl není úplně zřejmý, ale to je v pořádku.

VM (Pošta)

Ocitujme často kladené otázky (VM FAQ): VM (View Mail) je subsystém Emacsu, jehož prostřednictvím můžeme číst a zpracovávat poštu v rámci Emacsu. Příkazy jsou obdobné jako v běžném uživatelském agentovi, například genero-vání odpovědí, ukládání zpráv do složek, mazání zpráv atd. K dispozici je i celá řada dalších pří-kazů, jejichž pomocí můžeme provádět úlohy jako vytváření různých přehledů, přeposílání zpráv a prohlížení podle různých kritérií.
Když jsem začal používat Emacs, trochu jsem si VM vyzkoušel. Zjistil jsem, že je skvělou náhradou Pine, Elm a většiny jiných poštovních programů. Nechtěl jsem však používat na čtení zpráv a pošty různé programy. Vývoj VM neustále pokračuje a má dobrou podporu.

K dispozici je na: *http://www.wonderworks.com/vm/*.

Gnus (Pošta a zprávy)

Ocitujeme manuál GNUS: „Gnus je laboratoř na čtení zpráv (news). Nechá nás nahlédnout do všeho, jako by to byla diskusní skupina. Jejím prostřednictvím můžeme číst poštu, prohledávat adresáře, využívat ftp – a dokonce i číst zprávy."

Účelem programu Gnus je vybavit uživatele pro čtení zpráv stejně, jako jsou vybaveni editorem Emacs pro zpracování textů. Neklade si žádná omezení v tom, co vše uživatel smí. Podněcuje uži-vatele k rozšiřování tak, aby vyhověl všem jejich požadavkům. Program nechce řídit uživatele; ten by měli využívat (nebo i zneužívat) k tomu, aby si dělali, co chtějí.

V současnosti využívám ke zpracování pošty i zpráv právě GNUS (jak jsem předeslal už shora). Vývoj programu GNUS také neustále pokračuje a má dobrou podporu. K dispozici je na:
http://www.gnus.org/.

BBDB (rolodex)

BBDB je „rafinovaná databáze velkého bratra" (*Big Brother Database*), program typu rolodex pro Emacs, který je kompatibilní s většinou poštovních balíků systému Emacs (včetně VM a GNUS). K dispozici je na:
http://pweb.netcom.com/~simmonmt/bbdb/index.html.

AucTeX (jiný režim pro TeX)

AucTeX je jiný režim pro editování souborů TeX. Ocitujme ze stránek AucTeX: „AUC TeX je rozšiřitelný balík, který podporuje psaní a formátování souborů TeX ve většině variant GNU Emacs. Podporuje různé balíky maker, včetně AMS TeX, LaTeX a TeXinfo."

K dispozici je na: *http://sunsite.auc.dk/auctex/*.

Jiné zdroje

V této části jsou uvedeny odkazy na knihy, diskusní skupiny, rozesílací seznamy a ostatní zdroje, kde nalezneme další informace o editoru Emacs.

Knihy

Existuje několik skutečně dobrých knih o Emacsu. Navíc, ve většině knížek o Linuxu a Unixu bývá pár kapitol věnováno editorům Emacs a vi.

Learning GNU Emacs

Autoři: Debra Cameron, Bill Rosenblatt, Eric S. Raymond. Vydavatel: O'Reilly &

Associates – *http://www.ora.com/*.

Komentář

Pravděpodobně nejlepší kniha pro začátečníky. Poté co se čtenář seznámí s návodem a pro

čte si často kladené otázky, poslouží mu tato kniha jako souhrnná a velmi čtivá učebnice.

Writing GNU Emacs Extensions

Autor: Bob Glickstein. Vydavatel: O'Reilly & Associates – *http://www.ora.com/.*

Komentář

Kniha je vhodná pro uživatele, kteří už nějaký čas používají Emacs a rozhodnou se, že si napíší svůj vlastní režim nebo nějaké pokročilejší zákaznické přizpůsobení. I když se kniha nepokouší naučit čtenáře Lisp, obsahuje krátký úvod do tohoto jazyka.

Programming in Emacs Lisp: An Introduction

Autor: Robert J. Chassell.

Ze souboru README: „Kniha je elementárním úvodem do programování v Emacsu pro ty, kteří

nejsou programátory a ani se o programování nezajímají, avšak chtěli by si přizpůsobit nebo roz
šířit své výpočetní nástroje.“

Celý manuál lze získat jako anonymní FTP na GNU FTP serveru: ftp://prep.ai.mit.edu/gnu/emacs/.

Komentář

Je to dobrý úvod do programování v jazyce Lisp v prostředí editoru Emacs – i pro neprogramátory.

The GNU Emacs Lisp Reference Manual

Autor: Richard Stallman. Vydavatel: The Free Software Foundation – *http://www.fsf.org/.*Celý manuál lze získat jako anonymní FTP na GNU FTP serveru: ftp://prep.ai.mit.edu/gnu/emacs/.

Komentář

Kniha je kompletním průvodcem programovacího jazyka Emacs Lisp.

GNU Emacs a Vim

Autor: Jan Polzer.

Vydavatel: Computer Press.

Z popisu publikace: „S touto knížkou můžete poznávat editory čtením od začátku do konce, nej

lepší službu však splní v roli praktické příručky, budete-li ji mít otevřenou na stole zároveň s jed-ním z těchto editorů na obrazovce – poslouží vám jako rychlá a spolehlivá nápověda v konkrét-ních situacích.“

Internetové stránky EMACSulation

EMACSulation je sloupek psaný Erikem Marsdenem, který se pravidelně objevuje v on-line maga-zínu Linux Gazette na adrese *http://www.linuxgazette.com/.* Poslední příspěvek nalezneme vždy na adrese *http://www.linuxgazette.com/issue39/marsden.html* a na konci sloupku jsou odkazy na předchozí články.

Diskusní skupiny

Zadáte-li vyhledávání diskusních skupin obsahujících řetězec „emacs“, určitě naleznete mnoho odkazů. Já sám jsem našel:

comp.emacs
comp.emacs.sources
gnu.emacs
gnu.emacs.bug
gnu.emacs.help
gnu.emacs.sources

Informace v češtině a slovenštině

http://emacs.ic.cz/
http://www.root.cz/clanky/emacs-viac-nez-len-editor/

http://www.root.cz/clanky/emacs-registry-a-bookmarky/

E-mailové konference

Na stránkách Free Software Foundation existuje konference pro GNU Emacs. Více informací nalez

nete na adrese *http://mail.gnu.org/mailman/listinfo/help-gnu-emacs*.Jediná konference věnovaná editoru Emacs, kterou prozatím znám, je NT-Emacs. Je určena těm,kteří používají Emacs ve verzi pro Microsoft Windows. Další informace naleznete na*http://www.cs.washington.edu/homes/voelker/ntemacs.html* v části FAQ (často kladené otázky).

Archiv Emacs Lisp

Ze souboru README archivu Emacs Lisp: „Archivy Emacs Lisp na ftp.cis.ohio-state.edu obsahují různé fragmenty a balíky kódu Emacs Lisp. Emacs Lisp je jazyk, který používáme k rozšíření funkcí editoru GNU Emacs, jehož autorem je spo-lečnost Free Software Foundation. I když distribuce GNU Emacs samotná obsahuje velké množ-ství kódu v Emacs Lisp, uživatelé si často píší vlastní balíky, které tvoří rozhraní na jiné systémy, vytvářejí lepší podporu pro jazyky, v nichž programují, přidávají do Emacsu nové funkce nebo mění implicitní chování editoru. Převážná část obsahu tohoto archivu byla napsána uživateli a distribuovaná veřejně po Internetu prostřednictvím konferencí info-emacs nebo info-gnu-emacs, případně prostřednictvím diskusních skupin comp.emacs, gnu.emacs, resp. gnu.emacs.sources.“ Archiv naleznete na anonymním FTP na adrese ftp://ftp.cis.ohio-state.edu/pub/emacs-lisp/.

> Poznámka
>
> Pokud mohu říci, Emacs Lisp Archive pomalu zastarává. Našel jsem v něm velmi málo nových (nebo aktualizovaných) balíků, i když vím, že existují. Určitě je však nalezneme v diskusní skupině comp.emacs.sources. (Opravte mě, pokud se mýlím.)

Multimédia v distribuci Fedora

Úvod

Existuje mnoho Zapovězených programů, jež jsou potřeba pro zprovoznění Java appletů nebo Flash animací v prohlížeči Mozilla, přehrávání MP3 souborů, přehrávání Quicktime/AVI/RealMe-ad/Windows Media formátů nebo přehrávání chráněných DVD a ke kterým vývojáři Fedory nechtějí uvolnit návod. Buď ze strachu před porušením amerického patentového zákona DMCA (Digital Millennium Copyright Act – nevztahuje se na ČR/SR) nebo z jiných estetických či politic-kých důvodů.

Tento návod shromažďuje na jednom místě většinu informací souvisejících s přehráváním multi-médií. Nejedná se však o obecný návod pro multimédia na Linuxu. Pokud by se o něj jednalo, pak by musel obsahovat stovky skvělých nástrojů a balíčků (počínaje programem GIMP a všemi jeho potomky, odnožemi a symbionty). Balíčky, jimž se tento článek zabývá, pokrývají pouze poli-ticky problémové oblasti, které ohrožují monopoly a straší právníky.

Velmi dobré informace, jak nakrmit vaši Fedoru, je také možné nalézt na stránkách *http://fedora-news.org/* a na neoficiálních stránkách často kladených dotazů (*http://www.fedorafaq.org/*). Před-pokladem tohoto návodu, který jej odlišuje od zmíněných webových stránek, je, že jste stejně líní jako já, tj. chcete nainstalovat *Zapovězené programy* (a později je udržovat aktuální) pomocí stan-dardních nástrojů pro správu softwaru bez studování specializovaných textů, stahování zdrojových kódů nebo pouštění se do unikátních postupů na tvorbu potřebných nástrojů.

> Právnická poznámka
>
> Žádný zdrojový kód nebo odkaz na zdrojový kód jakéhokoli softwaru, který údajně zapadá pod DMCA, není uveden v tomto textu. Je možné je ale najít na mých soukromých stránkách. DMCA je špatný zákon bláznů a darebáků, kteří se snaží utišit váš hlas, a jehož hanebným úkolem je potlačit svobodu slova a svobodný software. Při psaní následujících postupů jsem měl ale tento zákon na paměti, aby se mohl návod co nejvíce rozšířit.

Jedním z důvodů, proč píši tento návod zrovna já, může být, že věřím, že nejsem snadný cíl pro útoky právníků, na rozdíl od většiny ostatních zkušených uživatelů. Publicita a repu-tace díky pravdomluvnosti zde velmi pomáhá. Pokud jste právník zvažující žalobu, pak vězte, že budu vytrvale vracet útok, protože jsem dobře znalý práva a velmi dobrý v komunikaci s tiskem, a vynaložím veškeré úsilí, abych vystavil vás a vašeho klienta veřejné ostu-dě za snahu o potlačování svobodného slova. Byli jste varováni.

Nové verze návodu

Nejnovější verze tohoto návodu je vždy k dispozici na webových stránkách *http://tldp.org/ HOWTO/Fedora-Multimedia-Installation-HOWTO/.*

Nemusíte se bát mi napsat jakýkoli komentář, dodatek nebo opravu chyb na Eric S. Raymond, (*esr@snark.thyrsus.com*). Prosím ale, abyste nepsali žádosti o radu či kvůli problémům s konfigu-rací. Takové zprávy budu ignorovat. Vše, co vím o tomto tématu, je již obsaženo v následujícím textu.

Balíčky, nástroje a repozitáře

Moderní linuxové distribuce si rychle zvykly fungovat způsobem, kdy fyzická média jsou použí-vána pouze na instalaci operačního systému a aktualizace jsou stahovány a instalovány z repozi-tářů umístěných na Internetu. Musíte vědět alespoň něco málo o těchto nástrojích a repozitářích, ze kterých se binární balíčky stahují.

Balíčky pro Fedoru se nazývají RPM. Každý může být instalován a odebírán samostatně a obsahuje nějaký program se všemi soubory, které potřebuje. Některé balíčky závisejí na jiných, například apli-kace vyžaduje pro svůj běh nějakou knihovnu. Jednou z věcí, kterou správce balíčků provádí, je sle-dování těchto závislostí a instalace potřebných balíčků dříve než původně požadovaný balíček.

Balíčky RPM naneštěstí mohou vzájemně kolidovat – například když chcete instalovat vzájemně nekompatibilní verze nějakého programu. Hlavním úkolem správce repozitáře je zajistit, aby balíč-ky RPM v jeho sbírce vzájemně nekolidovaly a aby všechny závislosti balíčků bylo možné vyřešit v rámci jeho repozitáře nebo nějakého obecného repozitáře, o kterém správce tvrdí, že jej využívá.

Existuje jeden hlavní repozitář pro Fedora Linux udržovaný samotným Fedora projektem a zhru-ba půl tuctu rozšiřujících repozitářů, které tento hlavní využívají. Rozšiřující repozitáře nebo repo-zitáře „třetích stran“ jsou přesně ty, kde je možné získat *Zapovězené programy*. Provozují je dob-rovolníci nezávisle na Fedora projektu, často mimo Spojené státy, kde nejsou vystaveni tamním predátorským právníkům.

Rozšiřující repozitáře vytvářejí zájmové skupiny rozdělené podle toho, kdo je s kým kompatibilní. Je možné pozorovat mnoho neshod a politických rozhodnutí mezi těmito skupinami, kterými se ale jakožto koncový uživatel prakticky nemusíte zabývat. Jediný důvod pro to, abyste o těchto sku-pinách věděli, je, že si musíte jednu z nich vybrat, abyste předešli případným kolizím balíčků. Počátkem roku 2006 bylo možné vybírat ze tří skupin, které si popíšeme později.

Nejdříve si musíme představit nástroje, díky kterým můžete získat *Zapovězené programy*:

yum (the Yellow Dog Updater, Modified) je řádkový klient, který je součástí distribuce Fedory. Pomůže vám stahovat aktualizace z repositářů Fedory a ostatních repositářů obsahující *Zapověze-né programy*, které Fedora neobsahuje.

Grafické rozhraní využívající yum, které bylo představeno ve Fedora Core 5. Jde o trochu přija-
telnější a snazší způsob, jak používat yum.Musíte také vědět něco o repozitářích. V každém z nich je sada balíků RPM, ale tyto jsou ještě roz-děleny na „podsady“ zvané „kanály“.

(http://fedora.redhat.com/)

Základní repozitář Fedory. Balíčky RPM obsažené v kanálech „stable“ a „updates“ (tedy „stabilní“ a „aktualizace“) jsou součástí instalačních médií.

(http://rpmforge.net/)

Skupina tvořená provozovateli většiny neoficiálních repozitářů vyjma Livna a ATrpms. Tyto balíky se zaměřují na rozšíření „Fedora Core“ a „extras“ repozitářů a jejich správci si dávají pozor, aby jejich balíky nekolidovaly s oficiálními balíky z repozitářů Fedory. Do skupiny patří *freshrpms*, *Dag Wieers*, *Dries*, *NewRPM* a *PlanetCCRMA*. Spolupracují na sjednocení jejich repositářů a již staví na společných zdrojových balících RPM. O RPMForge je ale bohužel známo, že balíky občas vážně kolidují s balíky z repozitářů Livna a ATrpms.

(http://rpm.livna.org/)

Repozitář umístění mimo Spojené státy a mimo dosah zákona DMCA, který je přímo zaměřený na poskytování *Zapovězených*

programů, které „Fedora Core“ a „extras“ neobsahují. Neexistuje mezi nimi žádné oficiální spojení a ve skutečnosti lidé z Fedory nezmiňují tento repositář na svých strán-kách nebo v dokumentaci ze strachu z napadení zákonem popírajícím svobodu slova nebo aso-ciací DVDCCA (DVD Copy Control Association). Lidé okolo Livna ale sledují Fedoru velmi důklad-ně, aby mohli balíky udržovat aktuální. Repozitář Livna závisí na oficiálních repozitářích Fedory. Kolidují s balíky z repositářů RPMForge.

(http://macromedia.mplug.org/)

Hlavní zdroj pro balíčky Macromedia Flash. Nekoliduje s žádnou skupinou repositářů.Tři skupiny, o kterých jsem se zmínil, jsou *RPMForge*, *livna* (sama o sobě) a *ATrpms* (sama
o sobě). Ke zprovoznění plné multimediální podpory ve Fedora Core 5 teoreticky stačí pouze repozitář Livna. Řekl jsem ‘teoreticky’, protože ve skutečnosti Fedora Core 5 v podstatě neobsa-huje žádnou podporu videa. V novějších verzích to bude podobné.

Přidání repositáře livna můžete provést následovně:

```
rpm -ivh http://rpm.livna.org/livna-release-5.rpm
```

Instalace by měla zanechat konfigurační soubor v adresáři /etc/yum.repos.d/.

Bezpečnostní a ostatní rizika

Všechny příkazy programu yum, které se objeví v tomto návodu, se musí spouštět pod uživate-lem root, aby balíčky, které stáhne, mohly být nainstalovány do vašeho systému. To znamená rizi-ko, že váš systém může být kompromitován balíčkem RPM obsahujícím trojského koně ať už ve formě balíčku obsaženého v některém dotazovaném repozitáři nebo balíčkem, který vám podstr-čí útočník mezi vámi a repozitářem pomocí útoku typu *man-in-the-middle*.

Pro omezení zmíněného rizika mnoho repozitářů poskytuje šifrované podpisy jejich balíčků RPM. Musíte mít nainstalovánu lokální kopii veřejného klíče z každého repozitáře, abyste mohli kontrolovat integritu stahovaných balíčků. Aktuální verze programu yum stáhne tyto klíče za vás. Klíče však mohou být ohroženy útokem typu man-in-the-middle, kdy vám útočník při vytváření vaší konfigurace podstrčí vlastní falešné klíče. Ačkoli je tato možnost extrémně nepravděpodobná, je dobré brát na ni ohled.

Dlouhodobé riziko, které přijímáte používáním proprietárního kódu odkazovaného z tohoto návo-du, je, že se stáváte závislým na rozmarech výrobce proprietárního softwaru. Není ani nutné uzná-vat staré náboženství Svobodného Softwaru, aby bylo jasné, že se jedná o problém. Některé pro-gramy zde zmíněné jsou volně distribuovány, ale s uzavřeným zdrojovým kódem – což je sice v pořádku, ale co se stane, až výrobce v budoucnosti změní své rozhodnutí? Ocitnete se v pasti.

Je nebezpečné stát se závislým na proprietárním softwaru a proprietárních formátech. Jakmile sám sobě dovolíte stát se na nich závislým, automaticky tak poškozujete ostatní pomáháním výrobcům udržovat nezdravý monopol na jejich oblast trhu. Pokud musíte investovat do těchto nástrojů, pro-sím, zkuste najít nějakou cestu, jak podporovat jejich náhrady s otevřeným zdrojovým kódem – přispějte vlastním vývojem nebo finančně, případně uzměte trochu svého času a tlačte na výrob-ce, aby své formáty otevřeli. Ukládejte své audio CD raději ve formátu Ogg Vorbis než MP3. Napiš-te dopis americkým zákonodárcům, aby zrušili zákon DMCA. Svoboda, kterou tím zachráníte, bude vaše vlastní.

Macromedia Flash

Fedora nedistribuuje Macromedia Flash, protože licence Macromedia to nepovoluje, ale neexistu

jí žádné právní překážky pro používání RPM z *http://macromedia.mplug.org/*.Podle návodu umístěte konfigurační soubor např. do adresáře /etc/yum.repos.d/. Soubor byměl vypadat následovně:

```
[macromedia] name=Macromedia for i386 Linux baseurl=http://macromedia.mplug.org/rpm/ enabled=1 gpgcheck=1
gpgkey=http://macromedia.mplug.org/FEDORA-GPG-KEY
```

Poté proveďte instalaci:

```
yum install flash-plugin
```

Instalace uvedeného RPM by měla vložit plugin do správných adresářů pro Firefox stejně jako pro

Mozillu.Můžete vyzkoušet funkčnost pluginu na oficiální zkušební stránce (*http://www.macromedia.com/shockwave/welcome/*). Pamatujte, že bude možná potřeba zavřít a znovu spustit váš prohlížeč. Jemožné, že Firefox spadne, pokud instalujete plugin ve chvíli, kdy je spuštěný.

Je tu však jeden chyták. Macromedia plugin funguje pouze na 32bitové platformě Intel. Na archi-tektuře x86_64 běžící v 64bitovém módu plugin vůbec nefunguje (obecně 32bitové pluginynefungují na 64bitových prohlížečích). V současnosti existují

3 projekty zaměřující se na tentoproblém:

gplflash1 (*http://gplflash.sourceforge.net/)* je původní Flash plugin uvolněný pod licencí GPL. Zvládá pouze soubory SWF do verze 4.
gplflash2 (*http://gplflash.sourceforge.net/*) funguje pouze z poloviny, s mnoha bolestivými nedostatky a častými pády Firefoxu. Tento pokus o přepsání gplflash byl zastaven ve pro-spěch projektu gnash.
gnash (*http://www.gnu.org/software/gnash/*) slibuje plnou podporu pro soubory SWF až do

verze 7, ale zatím se nachází v raném stadiu vývoje a její použití je na vlastní riziko. Jako dočasné řešení pro 64bitové systémy by mohla posloužit kompilace a instalace gplflash1.

gstreamer a podpora ffmpeg

Fedora Core 5 přijala projekt nazvaný *gstreamer* jako obecný nástroj pro zpracování video a audio souborů. Samostatný přehrávač multimédií *Totem* dodávaný společně s FC5 používá gstreamer, stejně jako plugin do Firefoxu *mozplugger*.

Sám gstreamer využívá pluginy, a to pro každý podporovaný formát jeden. Pluginy pro otevřené formáty jako Ogg Vorbis jsou součástí FC5. Pluginy pro proprietární formáty samozřejmě nikoli. Abyste je získali, použijte tento příkaz:

```
yum install gstreamer-plugins-ugly ffmpeg
```

Jako závislosti by se měly nainstalovat také balíčky mpeg2dec, libsidplay, libdvdread, faac, faad2,
imlib a gsm. Klíčové slovo „ugly“ (ošklivý, pozn. překladatele) v názvu balíčku gstreamer-plugins-ugly ozna-čuje sadu „nežádoucích“ pluginů zvládající formáty MP3, MPEG2 a SID. Formát SID je historická relikvie z dob Comodore 64. Formát MP3 je nejpopulárnější audio formát. MPEG2 je audio/video formát, který se používá pro DVD disky.

Knihovna ffmpeg umí kódovat nebo dekódovat audio a video ve formátech MPEG1, MPEG4, h263, ac3, asf, avi, real, mjpeg a flash.

MP3

Distribuce Fedora neobsahuje žádný software schopný přehrávat soubory typu MP3, protože
patentová licence Fraunhoferova institutu není kompatibilní s licencí GPL. Implicitní hudební přehrávač ve Fedora Core 5 je program *Rhythmbox.* Ztratil jsem s tímto pro-gramem několik hodin, ale i přes jeho hezký vzhled se ukázal být nepoužitelným. Ignoruje čísla skladeb – respektive nemá dostatek inteligence na získání čísel skladeb z názvů souborů, podob-ně jako *xmms*, a nikde nejsou dokumentovaná pravidla pro vkládání souborů. Pokusy o zpro-voznění podpory MP3 pro tento přehrávač se ukázaly být noční můrou tak strašidelnou, až jsem v jednu chvíli myslel, že budu muset přeinstalovat celý systém. A to jen proto, abych získal zpět alespoň podporu pro Ogg Vorbis, protože jsem nemohl zjistit, jaká část nedokumentované apli-kace se zhroutila a jak to opravit. Dokumentace je ve skutečnosti špatný vtip – útržkovitá, leda-bylá a bez jakýchkoli užitečných informací o řešení problémů. Přehrávač xmms možná nemá tolik funkcí a nevypadá tak hezky, ale funguje mnohem lépe.

Instalace xmms a podpory MP3 provedete následujícím příkazem:

```
yum install xmms xmms-mp3
```

Pro samotné přehrávání souborů MP3 musíte spustit xmms a upravit jeho konfiguraci. Vyberte z menu Options -> Preferences -> Audio I/O Plugins. Vyskočí okno se seznamem dostupných plu-ginů, ze kterého vyberte „MPEG Layer 1/2/3 Placeholder Plugin“ a odznačte „Enable Plugin“. Tím se náhradní plugin deaktivuje a xmms začne používat MP3 plugin automaticky. Bude potřeba re-startovat xmms.
Na mém počítači na platformě x86_64 jsem narazil na problém, kdy xmms přehrával hudební sou-bory, pouze pokud program běžel pod uživatel root. Pod jiným účtem se okamžitě automaticky ukončil. Může to být způsobeno mnohými rozličnými problémy. Zkuste zkontrolovat přístupová práva na zvukových zařízeních. Já jsem narazil na jiný exotický problém (*http://lists.xmms.org/ pipermail/xmms-devel/2000-January/001083.html*), který vyžadoval, aby zvukový modul byl nastaven na používání low-memory DMA.

Pro aktivaci podpory přehrávání souborů MP3 ve Firefoxu a Mozille potřebujete balíčky gstrea-mer-plugins-ugly a ffmpeg (viz výše). MP3 soubory budou přehrány v xmms, podcast přes Totem.

Java

Javu lze stáhnout přímo ze stránek společnosti Sun a posléze ji distribuovat, ale pouze pro osob-ní a nekomerční účely. Licence Javy od firmy Sun není open-source, a proto Fedora, stejně jako většina ostatních distribucí, tento balíček neobsahuje. Vzhledem k uvedení Javy pod GPL se to ale možná brzy změní.

Fedora Core 5 obsahuje open-source implementaci jazyka Java určenou pro vývojáře, která se nazývá *gcj*. Tento balíček nestačí pouze v případě, že chcete aktivovat Java applety ve vašem pro-hlížeči. Na neoficiálních stránkách často kladených dotazů Fedory je detailně popsáno, jak apple-ty zprovoznit (*http://www.fedorafaq.org/#java*).

Plugin můžete otestovat na zkušebních stránkách Java appletů firmy Sun (*http://www.java. sun.com/applets/*). Některé z těchto appletů ale v době, kdy jsem to zkoušel, nefungovaly úplně správně (Escher a Starfield), nicméně BouncingHeads se zdál být pro testy nejvhodnější.

Může se také hodit přesunutí hotových balíčků RPM, které jste vytvořili, někam mimo adresář /usr/src/redhat/RPMS. Po instalaci nového systému by mohly být užitečné znovu.

RealAudio a RealVideo

Momentálně nevím o žádném místě, kde by bylo možné stáhnout tyto balíčky přímo přes yum, takže jedinou možností je program stáhnout a nainstalovat ručně. Nejdříve je však nutné nainsta-lovat jiný balíček:

```
yum install compat-libstdc++-33
```

Poté můžete stáhnout balíček RPM RealPlayer10GOLD (*http://forms.real.com/real/realone/ intl/focus.html*) a nainstalovat jej příkazem:

```
rpm -Uvh RealPlayer10GOLD.rpm
```

Na 32bitové Fedora Core možná budete nejdříve muset odinstalovat přehrávač HelixPlayer příkazem:

```
rpm -e HelixPlayer
```

Fedora Core 5 pro architekturu x86_64 neobsahuje HelixPlayer.

MPEG, QuickTime, AVI a DVD

MPEG (formát používaný na DVD) je prezentován jako otevřený standard, ale většina linuxových distribucí neobsahuje software, který by jej uměl zpracovat, kvůli patentu drženému firmou MPEGLA. Stejně tak formáty AVI a Apple QuickTime obsahují proprietární patentované kodeky,
takže mnohé linuxové distribuce opět nedodávají software umožňující jejich dekódování. Fedora Core 5 obsahuje oficiální přehrávač z prostředí GNOME, který je nazvaný *Totem* (*http://www.gnome.org/projects/totem/*). Ten ale bohužel nepřehrává správně DVD. Problémy se projevují chybovým hlášením "Totem was not able to play this disc. No reason." (Totem nemohl přehrát tento disk. Důvod neuveden.), nicméně se jedná o známou chybu aplikace gstreamer verze 0-10.
Alternativní přehrávač je na tom někdy ještě hůř. Nainstalovat jej můžete příkazem:

```
yum install xine xine-lib libdvdcss
```

Tato akce nainstaluje ještě množství podpůrných knihoven, včetně *libdvdcss*, o které se ale vývojáři přehrávače xine na svých stránkách raději nezmiňují ze strachu před útoky právníků aso-ciace DVDCCA.

Na mém počítači (s vanilla jádrem pro Opteron a grafickou kartou nVidia GeForce2) xine verze
0.99.4 zobrazuje pouze bílé okno.

Testovací stránky pro online vysílání

Na následujících odkazech můžete vyzkoušet online přehrávání audio a video souborů:

AVI (*http://codeccorner.com/*).
QuickTime (*http://www.apple.com/quicktime/troubleshooting/*).
Windows Media (*http://www.vdat.com/techsupport/windowstest.asp*).
MPEG (*ftp://ftp.tek.com/tv/test/streams/Element/index.html*).
RealPlayer (*http://service.real.com/test/*).

Související zdroje

Průvodce snadnou instalací Fedora Core 5 včetně návodů ze stejné oblasti jako tento text můžete
nalézt na *http://stanton-finley.net/fedora_core_5_installation_notes.html.*Tipy a triky pro Fedora Core 4 jsou k dispozici na stránkách *http://home.gagme.com/ greg/linux/fc4-tips.php.*

Dobrý návod na instalaci multimediálních pluginů pro Firefox můžete nalézt na *http://www.*
yolinux.com/TUTORIALS/LinuxTutorialMozillaConfiguration.html.

Návod k zaváděným modulům jádra Linuxu

Předmluva

Toto je návod k zaváděným modulům jádra Linuxu (Linux loadable kernel modules, LKM). Je v něm popsáno, k čemu tyto moduly slouží, jak se používají a způsob jejich vytváření. V návodu jsou také zdokumentovány parametry a další podrobnosti související s používáním některých modulů.

Návod udržuje Bryan Henderson *(bryanh@giraffe-data.com*). Anglický originál tohoto návodu v aktuální verzi naleznete na stránkách Linux Documentation Project (*http://tldp.org/*). Tento dokument je zaměřen hlavně na Linux 2.4. Má však platnost i pro starší verze Linuxu. Čás-tečně jej lze použít i pro novější verze, avšak v úplnosti je autor hodlá popsat až v příštím vydá-ní tohoto dokumentu, viz kapitola 12. Ve verzi 2.5 došlo k určitým změnám, které tento dokument nezachycuje, neboť verze 2.5 nebude poskytnuta k užívání veřejnosti. Veškeré změny se promít-nou až do návodu k verzi 2.6, pokud do ní budou zahrnuty.

Copyright

Úvod do zaváděných modulů jádra Linuxu

Chcete-li do jádra Linuxu přidat nějaký kód, budete nejspíš postupovat tak, že ke zdrojovému stro-mu jádra přidáte další zdrojové soubory a jádro znovu přeložíte. Konfigurace jádra vlastně spočí-vá zejména ve výběru souborů, které mají být přeloženy společně s jádrem.

Kód lze ovšem přidat do jádra i v době jeho běhu. Takový kód nazýváme *zaváděný modul*. Může mít nejrůznější funkce, obvykle však je to jedna z těchto tří možností: 1) ovladač zařízení; 2) ovla-dač souborového systému; 3) systémové volání. Uvedené funkce, a ještě i některé další, mohou být prováděny odděleně a nemusí být složitě začleňovány do celého jádra.

Terminologie

Zaváděným modulům jádra často říkáme jenom moduly jádra, nebo dokonce jen moduly, což však je poněkud zavádějící název, neboť na světě existuje mnoho různých modulů a různé části kódu zabudované do základního jádra bychom také klidně mohli nazývat moduly. Pojmem zaváděný modul (zaváděný modul) označujeme určitý druh modulu, o němž pojednává tento návod.

> Poznámka
>
> Přestože má autor pravdu – pojmem modul lze opravdu označit jakýkoliv souvislý blok kódu – v češtině se výraz *zaváděný modul jádra* téměř nepoužívá. Daleko častější je ozna-čení *modul jádra* nebo *jaderný modul*. Označení zaváděný modul jádra jsme přesto pone-chali, protože je důležitý pro rozlišení některých vlastností dále v textu návodu, kde s ním původní autor počítá. Mějte tuto skutečnost na paměti při procházení internetových zdro-jů v češtině!

Existuje názor, že zaváděné moduly se nacházejí mimo jádro, a v této souvislosti se hovoří o komunikaci zaváděného modulu s jádrem. To je ovšem chyba. Zaváděný modul je po zavede-ní nedílnou součástí jádra. Správné označení obrazu části jádra, které zavádíte (tj. vše *kromě* zavá-děných modulů), je „základní jádro“. Zaváděný modul komunikuje se základním jádrem.

V některých operačních systémech se ekvivalent linuxového zaváděného modulu nazývá „rozší ření jádra“.
Co je tedy „Linux“? Nejdříve musíme konstatovat, že tímto jménem jsou označovány dvě zcela roz dílné věci, a zde budeme hovořit pouze o jedné z nich:

Jádro a příslušenství, které tvoří balík distribuovaný Linusem Torvaldsem.
Třída operačních systémů obvykle založených na jádru Linuxu (označení GNU/Linux).

Výklad o zaváděných modulech se vztahuje pouze k prvnímu z nich. I když však zavedeme tuto definici, dochází v souvislosti se zaváděnými moduly k určitým nejasnostem. Je zaváděný modul součástí Linuxu, nebo není? Zaváděný modul je součástí jádra, a tedy i Linuxu tehdy, když je sou-částí balíku distribuovaného jako jádro Linuxu, jinak ne. Pokud tedy do jádra zavedete ovladač zařízení, který obdržíte společně s tímto zařízením jako zaváděný modul, nelze, přesně řečeno, hovořit o tom, že jádro je stále ještě Linux. Spíše jde o Linux, který je určitým způsobem rozšíře-ný. Jak se dalo očekávat, tato situace vede k nepřesnému používání názvu „Linux“ – tímto názvem je označováno mnoho upravených operačních systémů typu Linux, které jsou ve značném počtu distribuovány. V zájmu přesnosti se budeme v tomto dokumentu držet co nejpřísnější definice.

Historie zaváděných modulů jádra

V začátcích Linuxu zaváděné moduly jádra neexistovaly. Veškeré funkce, které nyní řešíme pro-střednictvím zaváděných modulů, byly dříve součástí jádra už při jeho kompilaci. Zaváděné modu-ly se objevily až v Linuxu 1.2 (1995).

Ovladače zařízení apod. byly vždy vytvářeny modulárně. Když byly vynalezeny zaváděné moduly, přepsání vestavěných modulů do tvaru zaváděných modulů nevyžadovalo příliš velký objem práce. Musely se však přepsat úplně všechny moduly, což přece jen zabralo určitý čas. Kolem roku 2000 vše, co mohlo být zaváděným modulem jádra, bylo jím volitelně.

Kdy je vhodné použít zaváděný modul jádra

Většinou si můžete vybrat, zda z daného modulu vytvoříte zaváděný modul jádra anebo jej do základu jádra zabudujete. Zaváděný modul má oproti vestavěnému modulu řadu výhod a doporučuji použít toto řešení všude, kde je to možné.

Jednou z výhod je skutečnost, že není nutné tak často překládat jádro. Ušetříte tím čas a zmenší-te možnost zavlečení chyby do základního jádra při jeho opakovaném vytváření a instalaci. Je vhodné ponechat funkční jádro beze změny pokud možno co nejdéle.

Další výhodou je možnost využít zaváděné moduly k diagnostickým účelům. Chyba v ovladači, která by se stala součástí jádra, by například mohla zcela znemožnit zavedení systému. A bývá velice obtížné určit, která část základního jádra má tento stav na svědomí. Je-li naopak ovladač zaváděným modulem, jádro je v činnosti ještě před zavedením tohoto modulu. Havaruje-li systém až po spuštění jádra v průběhu další činnosti, je nalezení chyby v ovladači mnohem snazší a do té doby můžete systém provozovat bez ovladače.

Dále může zaváděný modul šetřit paměť, neboť jej stačí zavést až tehdy, když jej skutečně používáte, zatímco všechny části základního jádra jsou trvale zavedeny, a to ve skutečné paměti, nikoli jen ve virtuální.

Zaváděné moduly se mnohem snáze udržují a ladí. Kvůli jedné změně v ovladači není nutno znovu zavádět celý systém, stačí zadat několik příkazů, které jsou provedeny mnohem rychleji než nové zavedení systému. Můžete přitom vyzkoušet různé parametry, nebo dokonce průběžně upravovat kód, aniž byste museli neustále zavádět systém.

Zaváděné moduly mimochodem nejsou o nic pomalejší než vestavěné moduly, které jsou součástí základního jádra. V obou případech proběhne volání tak, že se řízení předá na příslušné místo v paměti.

Někdy je ovšem nutné zabudovat kód přímo do jádra systému. Jde zejména o činnosti, které je nutno provést před zaváděním modulů. Například ovladač diskové mechaniky, na níž je uložen kořenový systém souborů, musí být součástí základního jádra. Dnes je ovšem možné obejít i tuto podmínku omezení pomocí initrd, jak si ukážeme v kapitole „Zavádění systému bez ovladače disku“.

K čemu nejsou zaváděné moduly vhodné

Občas se můžeme setkat s názorem, že zaváděný modul jádra je něco jako uživatelský program. Oba typy programů skutečně mají řadu společných vlastností, avšak zaváděný modul v žádném případě není uživatelským programem, nýbrž je součástí jádra. Jako takový nepodléhá žádným systémovým omezením a snadno může způsobit havárii systému.

K čemu se používají

Existuje šest hlavních použití zaváděných modulů:

Ovladače zařízení. Ovladač je určen pro obsluhu určité součásti hardwaru počítače a jádro s tímto hardwarem komunikuje prostřednictvím tohoto ovladače, aniž by muselo znát jaké-koli podrobnosti o principech činnosti tohoto hardwaru. Existuje například ovladač mecha-niky ATA disku a jiný ovladač pro karty NE2000 a kompatibilní. Oba ovladače jsou sou-částí jádra proto, aby mohlo komunikovat s těmito zařízeními.
Ovladače souborových systémů. Ovladač souborového systému interpretuje souborový systém jako soubory, adresáře atd. Existuje mnoho jiných způsobů, jak ukládat soubory, adresáře apod. na diskovou mechaniku, na síťové servery i jinam. Pro každý způsob uklá-dání však potřebujete ovladač souborového systému. Existuje například ovladač souboro-vého systému ext2, který se v Linuxu používá téměř výhradně. Jiný ovladač je v souboru MS-DOS, jiný například v NFS.
Systémová volání. Uživatelský program používá systémová volání, když požaduje po jádru nějaké služby. Existují například systémová volání pro čtení souboru, pro vytvoření nové-ho procesu nebo pro ukončení činnosti systému. Většina volání je integrální součástí syste-mu a jsou standardizovaná, takže jsou vždy vestavěná do základního jádra (nikoli jako zaváděný modul). Můžete si však vymyslet svoje vlastní systémové volání a nainstalovat je jako zaváděný modul. Anebo se vám nelíbí, jak Linux provádí některá systémová volání a přepíšete je svým vlastním voláním zaváděných modulů.
Síťové ovladače. Síťový ovladač interpretuje síťový protokol. Odesílá data různým vrstvám a přijímá data z různých vrstev síťových funkcí jádra. Když například chcete vytvořit síťo-vé připojení IPIX, musíte použít ovladač IPIX.
Řádkové terminály. V zásadě jde o přírůstky ovladačů terminálových zařízení.
Interprety spustitelných souborů. Interpret spustitelného souboru slouží k zavádění a spouš-tění spustitelného souboru. V Linuxu je možné spouštět spustitelné soubory různých for-mátů a každý musí mít vlastní interpret.

> Poznámka
>
> Puntičkáři nechť nahlédnou do kapitoly „Přidělování paměti při zavádění“.

Jak se dělají zaváděné moduly jádra

Zaváděné moduly jádra jsou uloženy v souborech typu ELF (Executable and Linking Format) a obvykle mají jméno např. serial.o apod. Soubory s moduly jsou většinou uloženy ve zvlášt-ních adresářích (vhodné místo je někde poblíž obrazu základního jádra). V programu insmod, kterým vkládáte zaváděný modul do jádra, pak uvedete jméno tohoto souboru s modulem.

Kompilace zaváděných modulů, které jsou součástí Linuxu, probíhá v rámci stejného procesu jako kompilace jádra samotného a výsledkem je obraz základního jádra. Viz soubor README ve stromě se zdrojovým kódem Linuxu. Jen v krátkosti, jakmile vytvoříte obraz základního jádra příkazem např. make zImage, všechny zaváděné moduly vytvoříte příkazem:

```
make modules
```

Výsledkem je množina souborů zaváděných modulů (*.o) v celém zdrojovém stromě. (Ve star-ších verzích Linuxu bývaly v adresáři modules umístěném ve zdrojovém stromě symbolické odka-zy na všechny tyto soubory.) Jsou připraveny k zavedení, avšak pravděpodobně si je budete chtít nainstalovat do nějakého vhodného adresáře. Vhodné místo je popsáno v kapitole „Kde jsou ulo-ženy soubory zaváděných modulů?“ a zkopírujete je tam příkazem make modules_install.

Součástí konfigurace jádra Linuxu (v době kompilace) je výběr částí, které mají být v základním jádru, a částí, z nichž budou vygenerovány samostatné zaváděné moduly. V průběhu konfigurace (make config) jsou vám pokládány otázky zjišťující, kterou část chcete dát přímo do jádra (odpo-věď Y), z které chcete vytvořit zaváděný modul (odpověď M jako modul) a kterou část chcete vynechat (odpověď N). Ostatní konfigurační metody (make menuconfig, make xconfig) jsou podobné.

Jak je uvedeno v kapitole „Kdy je vhodné použít zaváděný modul jádra“, do základního jádra byste měli dát jen nejnutnější minimum modulů a vynechat pouze ty části, které zcela určitě nebudete potřebovat. A pokud některou z nich nevynecháte, nic moc se nestane. Bude vás to stát jen tro-chu času při překladu, trochu místa na disku a pravděpodobnost, že bude při kompilaci jádra něco chybět, se nepatrně sníží. Tak to je.

V průběhu konfiguračního dialogu také musíte zadat, zda budete používat symbolické značení verzí či nikoli. To ovlivní kompilaci jak základního jádra, tak i zaváděných modulů, což je rozhodnutí zcela zásadní, viz kapitolu „Nenalezené symboly“.

Zaváděné moduly, které nejsou součástí Linuxu (tj. nejsou distribuovány s jádrem Linuxu), mají vlastní procedury pro kompilaci, o kterých se zde nebudu zmiňovat. Snad jen tolik, že tyto pro-cedury slouží ke zpracování souborů ELF s moduly.

Zaváděné moduly a základní jádro nemusíte nutně kompilovat současně (tj. můžete provést gene-raci pouze základního jádra a použít zaváděné moduly, které jste vygenerovali někdy dříve), avšak tento postup lze doporučit, viz kapitolu „Vkládání a odstraňování zaváděných modulů“.

Nástroje pro práci se zaváděnými moduly

Programy pro zavádění, rušení a jiné úkony se zaváděnými moduly jsou v balíku modutils, kterýnajdete na adrese http://www.kernel.org/pub/linux/utils/kernel/modutils.Tento balík obsahuje následující programy, které usnadňují práci se zaváděnými

moduly:

insmod

Nahrej zaváděný modul do jádra.

rmmod

Odstraň zaváděný modul z jádra.

depmod

Urči závislosti mezi zaváděnými moduly.

kerneld

Démon jádra.

ksyms

Vypiš symboly exportované jádrem pro použití novými zaváděnými moduly.

lsmod

Vypiš seznam stávajících modulů, které jsou zavedeny.

modinfo

Vypiš obsah části .modinfo v ELF souboru zaváděného modulu.

modprobe

Inteligentně nahrej nebo odstraň zaváděný modul nebo množinu zaváděných modulů. Když je například nutno před modulem B zavést nejprve modul A a řeknete modprobe, aby zavedl modul B, automaticky nejdříve zavede modul A.

Změny v jádru s sebou často přinášejí i nutnost změn v modutils, musíte se tedy přesvědčit, že k aktualizovanému jádru používáte i správnou verzi modutils. Tento balík je vždy zpětně kompa-tibilní (tj. funguje i se staršími jádry), takže stačí mít nejnovější verzi modutils.

> Upozornění modprobe vyvolává insmod a jeho umístění má pevně zakódované jako /sbin/insmod. V modutils může být nastavena i jiná cesta, která se běžně nepoužívá k hledání programů ($PATH). Buďto tedy před kompilací modutils v něm změňte zdrojový kód nebo se pře-svědčte, že programy instalujete do obvyklých adresářů.

Vkládání a odstraňování zaváděných modulů

Základními programy pro vkládání a odstraňování zaváděných modulů jádra jsou insmoda rmmod. Podrobnosti naleznete v manuálových stránkách.Vložení zaváděného modulu jádra je formálně jednoduché: Jako superuživatel pouze zadáte příkaz:

```
insmod serial.o
```

(serial.o obsahuje ovladač sériových portů (UART)).Kdybych ovšem pouze konstatoval, že tento příkaz funguje, nebyla by to tak docela pravda. Často
dochází k takové nesmyslné věci, že se příkaz neprovede a vypíše se buď zpráva o tom, že nesou-hlasí verze jádra a modulů, anebo hromada nevyřešených symbolů. Pokud ovšem příkaz funguje, asi víte, co děláte, a můžete se o tom přesvědčit nahlédnutím do

/proc/modules, jak je popsáno v kapitole „Soubor /proc.modules“.Všimněte si, že příklady uvedené v této kapitole, jsou z Linuxu 2.4. Ve verzi Linuxu 2.6 je technika
zavádění modulů podstatně odlišná, což je zřejmé zejména ze skutečnosti, že soubory zaváděnýchmodulů mají příponu „.ko“, nikoli „.o“. Z uživatelského hlediska však rozdíly nejsou tak patrné.Nyní se podíváme na poněkud složitější vkládání. Zadáte-li:

```
insmod msdos.o
```

pravděpodobně se vám vypíše hromada chybových hlášení, např.:

```
msdos.o: unresolved symbol fat_date_unix2dos msdos.o: unresolved symbol
fat_add_cluster1 msdos.o: unresolved symbol fat_put_super ...
```

To proto, že v msdos.o jsou odkazy na vypsané externí symboly, avšak jádro je neobsahuje. Můžete se o tom přesvědčit příkazem:

```
cat /proc/ksyms
```

který vypíše všechny symboly exportované jádrem (tj. na něž se mohou navázat zaváděné moduly). Symbol „fat_date_unix2dos“ však v seznamu není.
(V Linuxu 2.6 není program /proc/ksyms. Nahrazuje ho /proc/kallsyms; tvar má jako výstup

programu nm: vyhledejte symboly označené „t“.)Jak jej do tohoto seznamu dostanete? Zavedete jiný modul, takový, v kterém jsou tyto symbolydefinované a vyexportované. V tomto případě je zaváděný modul v souboru fat.o. Zadejte tedy:

```
insmod fat.o
```

a uvidíte, že „fat_date_unix2dos“ je v /proc/ksyms. Nyní znovu zadejte:

```
insmod msdos.o
```

a už to bude fungovat. Podívejte se do /proc/modules a uvidíte, že oba moduly jsou zavedené a vzájemně na sobě závisí:

```
msdos 5632 0 (unused) fat 30400 0 [msdos]
```

Jak jsem věděl, že chybí právě fat.o? Chce to jen trochu důvtipu. Problém lze vyřešit i poněkud „neohrabanějším“ (berte s rezervou) způsobem pomocí depmod a modprobe, viz dále. Když tyto symboly vypadají nějak jako „fat_date_unix2dos_R83fb36a1“, problém může být poně-kud složitější než jen zavedení určitého modulu. Viz kapitola „Nenalezené symboly“. V téže kapi-tole také naleznete, jak postupovat, když nesouhlasí verze jádra a modulu.

Při vkládání zaváděného modulu jádra je také často nutné předat modulu nějaké parametry. Ovladač zařízení například musí znát adresu a IRQ zařízení, které má ovládat. Anebo síťový ovladač musí vědět, kolik chcete provést diagnostických sledování. Viz příklad:

```
insmod ne.o io=0x300 irq=11
```

Zde zavádím ovladač adaptéru ethernetu typu NE2000 a říkám mu, že má ovládat ethernetový

adaptér na V/V adrese 0x300, který generuje přerušení na IRQ 11.Zaváděné moduly nemají žádné standardní parametry a ani konvencí není v této oblasti mnoho.O parametrech, které předá insmod do zaváděného modulu, rozhoduje jeho autor. Z toho plyne,že jejich popis naleznete v dokumentaci k těmto modulům. Obecné informace o parametrechmodulů naleznete dále. Zaváděný modul odstraníte z jádra příkazem:

```
rmmod ne
```

Pro výpis zavedených modulů je určen příkaz lsmod, který však pouze vypíše obsah /proc/modules se záhlavím, takže by snad bylo lepší věštit z křišťálové koule...

Nemohu najít verzi jádra...

Obvyklou chybou je zkoušet vkládat soubor, který není zaváděným modulem jádra. Představte si, že například zakomponujete modul pro paměť USB do základního jádra a nevytvoříte z něho zavá-děný modul. Použijete k tomu soubor usbcore.o, který vypadá úplně jako zaváděný modul. Avšak pomocí insmod se vám nepodaří jej zavést.

Vypíše se vám tedy zpráva, že byste měli jádro zkonfigurovat tak, aby modul pro USB byl zaváděným modulem? Jistěže ne. Jste v Unixu, kde jsou vysvětlující zprávy považovány za známku slabosti. Vypíše se zpráva:

```
$ insmod usbcore.o usbcore.o: couldn’t find the kernel version this module was compiled for
```

v níž insmod jenom říká, že v souboru usbcore.o hledal informaci, kterou má každý legitimní zaváděný modul – jádro verze, k němuž byl zaváděný modul vytvořen – a nenašel ji. Nyní už víme, že ji nenalezl proto, že soubor není zaváděným modulem. Jak zjistit, co vlastně insmod v soubo-ru vidí a proč nepovažuje soubor za zaváděný modul, to se dočtete v kapitole „Sekce .modinfo“.

Je-li to modul, který jste vytvořili s úmyslem, aby byl zaváděným modulem jádra, musí následovat otázka: Proč jím není? Nejčastější příčinou je, že jste na začátek zdrojového kódu nedali #inclu-de <linux/module.h> a/nebo nedefinovali makro MODULE. Makro bude nastaveno příkazem kompilátoru (-DMODULE) a určí, zda překladem má vzniknout zaváděný modul jádra nebo vesta-věný modul, který se stane součástí jádra. Pokud má být modul *pouze* zaváděným modulem, jak je to v současné době obvyklé, musíte tuto skutečnost definovat ve zdrojovém kódu (#define MODULE) ještě dřív, než přidáte příkaz include/module.h.

Co se děje při zavádění modulu

Úspěšně jste zavedli modul a zavedení ověřili pomocí /proc/modules. Ale jak se dozvíte, zda fun-guje? To je úkol zaváděného modulu samotného a závisí na druhu modulu, zde však jsou někte-ré obvyklé činnosti, které modul provádí po zavedení.

První věc, kterou ovladač provede po zavedení (a co by provedl po zavedení systému, kdyby byl součástí základního jádra), je, že vyhledá zařízení, které umí ovládat. Způsob hledání je ovšem u každého ovladače jiný a obvykle jej lze řídit pomocí parametrů. Avšak v každém případě, pokud ovladač nenajde zařízení, které by mohl ovládat, způsobí havárii zavaděče. Jinak se zaregistruje jako ovladač daného hlavního čísla a můžete začít zařízení používat prostřednictvím speciálního souboru, který náleží k tomuto hlavnímu číslu. Také se může zaregistrovat jako program, který zpracovává přerušení na úrovni odpovídající tomuto zařízení.

Rovněž může vydávat příkazy, kterými zařízení nastaví, takže na něm může něco zablikat apod.

Registraci ovladače si můžete ověřit v souboru /proc/devices a zpracování přerušení generovaná tímto zařízením v souboru /proc/interrupts. Dobrý ovladač navíc vydává zprávy jádra o tom, které zařízení našel a že je připraven k činnosti.(Ve většině systémů jsou zprávy jádra směrovány na konzolu a do souboru /var/log/messages.Předchozí zprávy si můžete zobrazit pomocí programu dmesg.) Některé ovladače však jsou tiché.Dobrý ovladač také předává (jako zprávy jádra) podrobnosti o průběhu vyhledávání, když zaří-zení nenalezne, avšak mnohé ovladače pouze způsobí havárii zavaděče bez jakéhokoli vysvětle-ní a k dispozici budete mít pouze seznam pravděpodobných příčin havárie vytvořený programeminsmod.

Ovladač síťového zařízení (rozhraní) pracuje podobně s tím rozdílem, že zaváděný modul regis-truje jméno zařízení (např. eth0), nikoli hlavní číslo. Jména zaregistrovaných síťových zařízení uvi-díme v souboru /proc/net/dev.

Ovladač souborového systému je při zavedení zaregistrován jako ovladač typu souborového systé-mu určitého jména. Například ovladač msdos je zaregistrován jako ovladač souborového systému typu msdos. (Tvůrci zaváděných modulů jim obvykle dávají jména shodná s názvy typů souboro-vých systémů, které budou obsluhovat.)

Inteligentní zavádění modulů – modprobe

Jakmile si vytvoříte představu o zavádění a rušení modulů pomocí programů insmod a rmmod, můžete začít používat program vyšší úrovně modprobe, čímž přesunete většinu práce se zavá-děním na systém. Podrobnosti viz manuálové stránky modprobe. Hlavní věc, kterou modprobe provádí, je automatické zavádění modulů, které zadaný zaváděný modul potřebuje ke své činnosti. Využívá k tomu soubor, jejž musíte sami vytvořit pomocí pro-gramu depmod a mít jej v systému.

Příklad:

modprobe msdos

Pomocí programu insmod zavede msdos.o, avšak ještě dříve (opět pomocí insmod) zavede fat.o, neboť ten musí být zaveden před msdos.o. Druhou důležitou věcí, kterou modprobe dělá, je vyhledání souboru, který obsahuje zaváděný modul zadaný pouze jménem. Například příkazem modprobe msdos můžete zavést /lib/2.4.2-2/fs/msdos.o. Ve skutečnosti mohou být parametry modprobe zcela symbolická jména přiřazená některému skutečnému modulu. Například příkaz modprobe eth0 zavede pří-slušný ovladač zařízení, který bude obsluhovat zařízení eth0 za předpokladu, že jste v modu-les.conf provedli příslušné nastavení. Podrobnosti o modprobe a o konfiguraci vyhledávacích pravidel v souboru modules.conf (obvykle v /etc/modules.conf, v jádrech řady 2.6 je to /etc/modprobe.conf) naleznete v manuálových stránkách modprobe.

Program modprobe je zvláště důležitý zejména proto, že je implicitním programem, který použí-vá zavaděč jádra k zavádění modulů. Budete-li používat automatické zavádění modulů, je nutno správně nastavit soubor modules.conf, jinak nebude fungovat. Viz kapitolu „Automatické zavá-dění a rušení modulů“.

Program depmod skenuje soubory zaváděných modulů (obvykle všechny soubory .o v poda-dresáři /lib/modules) a zjišťuje, které moduly potřebují ke své činnosti jiné moduly (odkazují se na symboly, jež obsahují). Vytváří soubor závislostí (obvykle jménem modules.dep), který bývá pro modprobe potřebu uložen v adresáři /lib/modules.
Program modprobe je také možno využít k odstranění zaváděných modulů. Pomocí konfiguračního souboru zaváděných modulů (obvykle /etc/modules.conf, /etc/mod-probe.conf) můžete jemně vyladit závislosti a ostatní náležitosti, které určují výběr zaváděných modulů. A také můžete specifikovat programy, které mají být spuštěny, když přidáte nebo odstra-níte zaváděný modul, například kvůli inicializaci ovladače. Když udržujete systém a dojde k tomu, že není k dispozici dostatek paměti, je zřejmě jednodušší nepoužívat modprobe a soubory a adresáře, které využívá, a dát do startovacích skriptů pouze insmod.

Automatické zavádění a rušení modulů Automatické zavádění

Pokud jádro potřebuje, aby byly některé moduly zaváděny automaticky, lze systém takto nastavit. Zavádění lze provádět buď zavaděčem modulů jádra, který je součástí jádra Linuxu, nebo jeho star-ší verzí, tj. démonem kerneld.

Nechť například běží program, který provádí otevřené systémové volání souboru v souborovém systému MS-DOS. Nemáte však ovladač ani v základním jádru systému, ani jako zavedený modul. Systém tedy neví, jak má otevřít soubor na disku.

Jádro rozpozná, že nemá ovladač souborového systému MS-DOS, avšak může použít jednu ze dvou možností automatického zavádění, čehož využije k zavedení daného modulu. Jádro pak dokončí otvírání souboru.

Automatické zavádění modulů jádra nemá pro složitost tohoto procesu smysl používat v moder ních systémech. Může to mít opodstatnění v malých systémech s nedostatkem paměti, protoženěkteré části jádra mohou být v paměti jen po dobu činnosti. Avšak velikost paměti, kterou tytomoduly ke své činnosti potřebují, je cenově natolik zanedbatelná, že je mnohem lepší pomocí star-tovacích skriptů při spouštění systému všechny potřebné moduly zavést a ponechat je zavedené.

Systém Red Hat Linux i další distribuce jako openSUSE nebo Mandriva Linux provádí automatic

ké zavádění modulů pomocí zavaděče modulů jádra.Zavaděč modulů i démon kerneld používají k vkládání zaváděných modulů modprobe, resp.insmod, viz kapitolu „Inteligentní zavádění modulů – modprobe“.

Zavaděč modulů jádra

Částečnou dokumentaci zavaděče modulů jádra naleznete v souboru Documentation/kmod.txt ve zdrojovém kódu Linuxu 2.4. Tato kapitola je ovšem úplnější a přesnější než uvedený soubor. Také můžete nahlédnout do zdrojového kódu v souboru kernel/kmod.c.

Zavaděč modulů jádra je volitelnou součástí jádra Linuxu. Do jádra jej lze zahrnout pomocí volby CONFIG_KMOD při jeho konfiguraci v průběhu kompilace. Když má jádro se zavaděčem zavést modul, vytvoří uživatelský proces (jehož vlastníkem je super-uživatel), který spustí modprobe, jenž modul zavede a ukončí se. Implicitně spouští modprobe ze souboru /sbin/modprobe, avšak to lze změnit zápisem jiného jména do souboru /proc/sys/kernel/modprobe, například:

```
# echo “sbin/mymodprobe” >/proc/sys/kernel/modprobe
```

Zavaděč modulů jádra předá modprobe tyto parametry: Nultý parametr je úplné jméno souboru modprobe. Normální parametry jsou -s, -k a jméno zaváděného modulu požadovaného jádrem. Parametr -s je „uživatelsky nepřátelský“ tvar --syslog, -k je zašifrovaný způsob, jak říct --auto-clean. Tj. zprávy z modprobe půjdou na démon syslog a zavedený modul bude mít nastavený příznak „autoclean“.

Nejdůležitější částí volání modprobe je pochopitelně jméno modulu. Všimněte si, že parametr „modul name“ v modprobe nemusí nutně být skutečné jméno modulu. Často to bývá pouze sym-bolické označení toho, co je úlohou modulu, a teprve v souboru modules.conf řeknete příkazem alias, který modul má být zaveden. Když je například obslužným modulem ethernetového adap-téru modul 3c59x, v souboru /etc/modules.conf budete pravděpodobně muset mít řádek:

```
alias eth0 3c59x
```

Podívejte se na některá z oblíbených jmen zaváděných modulů používaná zavaděčem (existuje asi 20 případů, kdy jádro volá zavaděč kvůli zavedení modulu):

Když se pokusíte použít zařízení a k hlavnímu číslu tohoto zařízení není zaregistrovaný žádný ovladač, jádro si vyžádá modul pomocí jména block-major-N nebo char-major-N, kde N je hlavní číslo v dekadickém tvaru bez levostranných nul.
Když se pokusíte použít síťové rozhraní (pravděpodobně programem ifconfig) a k roz-hraní není zaregistrovaný žádný ovladač, jádro si vyžádá modul stejného jména jako jméno rozhraní (např. eth0). To platí i pro ovladače logických (nikoli fyzických) rozhraní, jako např. ppp0.
Když zkusíte kontaktovat soket pomocí rodiny protokolů, která nemá zaregistrovaný ovla-dač, jádro si vyžádá modul jménem net-pf-N, kde N je číslo rodiny protokolů (dekadicky bez úvodních nul).
Když zkusíte exportovat adresář na NFS nebo prostřednictvím systémového volání jinak kontaktovat server NFS, jádro si vyžádá modul jménem nfsd.
Ovladač ATA (jménem ide) zavede příslušné ovladače pro třídy zařízení ATA jmény: ide-disk, ide-cd, ide-floppy, ide-tape a ide-scsi.

Zavaděč modulů jádra spustí modprobe s následujícími vnějšími proměnnými (pouze): HOME=/; TERM=linux; PATH=/sbin:/usr/sbin:/bin:/usr/bin.Zavaděč modulů jádra existuje od verze Linuxu 2.2 a byl navržen jako náhrada kerneld. Nestalse jí, avšak má všechny vlastnosti kerneld.

V Linuxu 2.2 vytváří zavaděč modulů jádra shora zmíněný proces přímo. V Linuxu 2.4 pracuje

zavaděč modulů jádra pro démona Keventd a je jeho dceřiným procesem. Zavaděč modulů jádra je dost podivný pavouk. Znásilňuje vnořování procesů proti všem pravi-dlům a zvykům v Unixu a v důsledku toho je nepružný, složitý, nesrozumitelný a málo stabilní.Mnoho systémových projektantů se naježí už jenom kvůli tomu, že má pevně nastavenou vyhle-dávací cestu. Klidně můžete místo tohoto zavaděče používat démona kerneld – nebo se alespoňnetrápit s automatickým zaváděním modulů.

Kerneld

Ve větším rozsahu je démon kerneld popsán v Kerneld mini-HOWTO, který naleznete v anglic-kém originále na adrese Dokumentačního projektu: http://www.tldp.org/. kerneld je uživatelský proces, který spouští program kerneld z balíku modutils. Do kanálu pro komunikaci mezi procesy (IPC message channel) přidá jádro. Když jádro potřebuje zaváděný modul, odešle do tohoto kanálu zprávu procesu kerneld, který spustí modprobe, aby zavedl modul, a pošle zprávu jádru, že je hotov.

Automatické rušení – automatické ukončení

Příznak automatického ukončení

Každý zavedený modul má příznak automatického ukončení, který může a nemusí být nastaven. Tento parametr ovládáte pomocí parametrů systémového volání init_module. Za předpokladu, že to děláte programem insmod, použijete volbu --autoclean.

Stav příznaku automatického ukončení naleznete v souboru /proc/modules. Modul, který má tento příznak nastavený, obsahuje i

příslušné vysvětlivky.

Odstraňování modulů s příznakem automatického ukončení

Účelem příznaku automatického ukončení je umožnit automatické odstranění modulu, který se po určitou dobu nepoužívá (obvykle 1 minutu). Tak pomocí automatického zavádění a rušení může-te mít zavedeny pouze ty části jádra, které jsou momentálně v činnosti, čímž šetříte paměť.

Tento způsob úspory paměti už není tak důležitý jako dřív, neboť paměť je nyní daleko levnější. Pokud nutně nepotřebujete šetřit paměť, je zbytečné se obtěžovat složitým zaváděním modulů. V inicializačním skriptu normálně zaveďte všechny moduly, které byste mohli potřebovat, a nechte je zavedené.

Existuje systémové volání delete_module, které říká „odstraň všechny zaváděné moduly, které mají nastaven příznak automatického ukončení a po určitou dobu nebyly v činnosti“. Program kerneld obvykle vydává toto volání jednou za minutu. Explicitně je můžete zadat příkazem rmmod --all.

Zavaděč modulů jádra odstraňování neprovádí. Pokud jej používáte, můžete si vytvořit opakovaně spouštěnou úlohu (cron), která bude tyto moduly pravidelně odstraňovat.

Soubor /proc/modules

Momentálně zavedené moduly zjistíte příkazem:
cat /proc/modules

který vypíše řádek (například):

serial 24484 0

Levý sloupec je jméno modulu, což je obvykle jméno souboru, z něhož jste tento modul zavedli, bez přípony „o“ (ko). Může obsahovat libovolné jméno, které bylo obsaženo ve volbě insmod.
Číslo „24484“ je velikost zaváděného modulu v bajtech. Číslo „0“ je čítač použití. Říká, kolik objektů momentálně na tomto modulu závisí. Typickými objekty jsou otevřená zařízení nebo připojené souborové systémy. Tento údaj je důležitý proto, že nelze odstranit zaváděný modul, pokud tento čítač není roven nule. Čítač si udržuje modul samot-ný, avšak správce modulů jej využívá k rozhodování, zda může povolit odstranění.

K popisu čítače shora existuje výjimka. Ve sloupci pro čítač se může objevit i -1, což znamená, že zaváděný modul tento čítač nepoužívá. Místo toho má zaregistrovaný podprogram, který může být volán správcem modulů a který vrátí příznak, zda může být zrušen. V takovém případě by mělo k zaváděnému modulu existovat uživatelské rozhraní a dokumentace s popisem, kdy je možné modul odstranit.
Pojmy „uživatelský čítač“ a „závislosti“ (popsané dále) nelze směšovat. Zde je jiný příklad s více informacemi:

```
lp              5280   0 (unused)
parport_pc      7552   1
parport         7600   1 [lp parport_pc]
```

Údaj v hranatých závorkách („[lp parport_pc]“) popisuje závislosti. Moduly lp a parport_pc se odkazují na adresy uvnitř modulu parport (prostřednictvím externích symbolů exportovaných programem parport). Moduly lp a parport_pc jsou tedy „závislé“ na modulu parport (modul parport obsahuje „závislost“).
Moduly se závislostmi nelze zrušit, avšak zrušením závislých modulů je možné tyto závislosti zrušit. Vysvětlivka „(unused)“ znamená, že tento modul nebyl nikdy použit, tj. nebyl nikdy ve stavu, kdy by nemohl být zrušen. Tuto informaci jádro uchovává z jednoduchého důvodu: Využívá ji při auto-matickém rušení modulů. V systému, v němž jsou moduly zaváděny a rušeny automaticky (viz
kapitola „Automatické zavádění a rušení modulů“), by se mohlo stát, že by si někdo zavedl modul, a než by jej stačil použít, byl by mu automaticky odstraněn, protože jej právě nikdo nepoužívá. Tento výpis normálně neuvidíte:

mydriver 8154 0 (deleted)

Modul je ve stavu „zrušený“. Jde ovšem o nepřesné vyjádření – ve skutečnosti právě probíhá ruše-ní tohoto modulu. (Pozn. překl.: Tato komplikace není ani tak způsobena neexistencí slovesných vidů v angličtině, jako spíš shora zmíněnou zásadou, že v Unixu je poskytování řádného vysvět-lení v chybových zprávách známkou slabosti. Nemůžete už zavést modul, který by na rušeném modulu závisel, avšak ten je ještě v systému. Zrušení modulu trvá jen malý okamžik, takže setká-te-li se s takovým stavem, modul bude zřejmě v nějakém neobvyklém stavu. Při ukončování se třeba mohl zacyklit, zůstal někde trčet nebo zhavaroval (čímž podtrhl jádru kobereček). V podob-ném případě je jedinou možností nové spuštění systému.

Existují podobné stavy „initializing“ a „uninitialized“, tj. počáteční nastavování a rušení počátečního nastavení.
Popisek „autoclean“ se vztahuje k příznaku automatického ukončení modulu, viz kapitolu „Auto-matické zavádění a rušení

modulů“.

Kde jsou uloženy soubory zaváděných modulů?

Svět zaváděných modulů je natolik flexibilní, že soubory s těmito moduly mohou žít v systému prakticky kdekoli. Existuje však určitá konvence, kterou většina systémů dodržuje: Soubory se zavá-děnými moduly typu .o jsou v adresáři /lib/modules, který je rozdělen na podadresáře. Pro kaž-dou verzi jádra existuje jeden podadresář, neboť moduly jsou závislé na verzích jádra (viz kapitola „Nenalezené symboly“). Každý podadresář obsahuje kompletní množinu zaváděných modulů.

Jméno podadresáře je rovno hodnotě výstupu z příkazu uname --release, například 2.2.19.
Podrobnější popis viz kapitola „Provoz několika jader“.Kompilujete-li Linux standardním způsobem, tedy pomocí příkazů make modules a makemodules_install, měly by být z příslušného adresáře nainstalovány všechny zaváděné moduly,jež jsou součástí dané verze Linuxu.

Kompilujete-li více jader, je vhodný jiný postup: Mějte uloženy moduly společně se základním jád-rem a s ostatními soubory, které náleží k jádru, v podadresáři adresáře /boot. Jedinou nevýho-dou je, že nemůžete mít adresář /boot uložen v malé oblasti na disku. V některých systémech je totiž /boot uložen ve zvláštní malé „zaváděcí oblasti“ (na speciálním oddílu) a obsahuje pouze ty soubory, jež provedou částečné spuštění systému do fáze, kdy k němu lze připojit jiný soubo-rový systém.

Nenalezené symboly

Obvyklou a velmi skličující chybou při zavádění modulů je chybové hlášení se seznamem nenalezených (unresolved) symbolů, např.:

msdos.o: unresolved symbol fat_date_unix2dos msdos.o: unresolved symbol fat_add_cluster1
msdos.o: unresolved symbol fat_put_super ...

Takto může skončit zavádění modulu z mnoha důvodů. V každém případě je nejdříve nutno nahlédnout do souboru /proc/ksyms a přesvědčit se, že vypsané symboly v něm skutečně nejsou.

Některé moduly ke své činnosti potřebují jiné moduly

Jednou z příčin může být skutečnost, že jste nezavedli jiný zaváděný modul, který obsahuje instrukce nebo data nutná k činnosti daného modulu. Toto se dá jednoduše vyřešit použitím pří-kazu modprobe, viz kapitola „Inteligentní zavádění modulů – modprobe“.

Modul musí odpovídat základnímu jádru

Tvůrci zaváděných modulů jádra si uvědomili, že v důsledku rozdělení jádra do několika soubo-rů (navíc do souborů distribuovaných nezávisle) může vzniknout problém. Co se stane, když modul mydriver.o byl napsán a přeložen tak, aby spolupracoval se základním jádrem 1.2.1, a někdo jej zkusí zavést do jádra 1.2.2? Co když mezi 1.2.1 a 1.2.2 nastala pouze malá změna, takže volání mydriver.o fungují? Jsou to interní podprogramy jádra, co kdyby se tedy od jedné verze k druhé neměnily? Bylo by po problémech.

Tvůrci se s touto situací vypořádali tak, že vybavili zaváděné moduly číslem jádra. Soubor my-driver.o má v sobě zvláštní sekci .modinfo, která v tomto příkladu obsahuje údaj „1.2.1“, neboť byla přeložena s hlavičkovým souborem z Linuxu 1.2.1. Kdybyste jej zkusili zavést do jádra 1.2.2,
program insmod ohlásí neshodu a ukončí se s tím, že nesouhlasí verze. Avšak moment. Jaká je skutečně pravděpodobnost toho, že mezi verzemi Linuxu 1.2.1 a 1.2.2 vznikla nekompatibilita, která bude mít vliv na modul mydriver.o? Ten pouze volá několik pod-programů a používá některé datové struktury, které se určitě nemění s každou novou verzí nej-nižší úrovně. Musíme znovu překládat všechny zaváděné moduly s hlavičkovými soubory těch jader, do nichž chceme tyto moduly vkládat?

Tuto komplikaci tvůrci odstranili tak, že k programu insmod přidali volbu -f, která „přinutí“ ins-mod, aby neshodu s verzí jádra ignoroval a modul vložil i tak. Vzhledem k tomu, že k význam-ným změnám verzí dochází velice zřídka, doporučuji používat -f vždy. Sice obdržíte varovné hlá-šení o neshodě, avšak zavádění pokračuje dál.

Tvůrci zaváděných modulů si nicméně uvědomovali, že k nekompatibilním změnám občas dochá-zí. Vymysleli proto velmi důmyslný způsob, jak učinit proces vkládání modulů citlivým na sku-tečný obsah všech rutin (podprogramů) jádra, které modul používá. Tento způsob se nazývá sym-bolické značení verzí (nebo poněkud nesrozumitelně „modulové značení verzí“). Je volitelné a zadáte jej při konfiguraci jádra uvedením volby „CONFIG_MODVERSIONS“.

Když kompilujete základní jádro nebo modul se symbolickým značením verzí, symboly exporto-vané pro použití v modulech jsou definovány jako makro. Definice makra má stejné jméno jako daný symbol a je k němu přidaná transformovaná hodnota parametru a typu návratové hodnoty daného podprogramu (založená na analýze zdrojového kódu podprogramu programem genk-syms). Podívejme se na podprogram register_chrdev, což je podprogram v základním jádru, který je často volán ovladačem. Při

používání symbolického značení verzí vypadá definice makra v jazyce C takto:

```
#define register_chrdev register_chrdev_Rc8dc8350
```

Tato definice platí jak ve zdrojovém souboru v C, který obsahuje podprogram register_chrdev, tak i ve všech zdrojových souborech, které se na tento podprogram odkazují. Zatímco vy při čtení kódu vidíte register_chrdev, preprocesor jazyka C ví, že tato funkce se ve skutečnosti jmenuje register_chrdev_Rc8dc8350.

Jaký má význam taková nesmyslná přípona? Je transformací datového typu parametrů a návrato-vé hodnoty podprogramu register_chrdev. Žádné dvě kombinace parametru a typu návratové hodnoty nemají stejnou transformovanou hodnotu.

Řekněme tedy, že někdo mezi verzemi Linuxu 1.2.1 a 1.2.2 přidá do register_chrdev parame-tr. Ve verzi 1.2.1 je makro register_chrdev definováno jako register_chrdev_Rc8dc8350, avšak ve verzi 1.2.2 jako register_chrdev_R12f8dc01. V souboru mydriver.o přeloženém s hlavičkovými soubory Linuxu 1.2.1 je externí odkaz na register_chrdev_Rc8dc8350, avšak základní jádro 1.2.2 takový symbol neexportuje. Místo něj exportuje symbol register_chr-dev_R12f8dc01.

Když se tedy pokusíte do základního jádra 1.2.2 vsunout mydriver.o odpovídající verzi 1.2.1, nepodaří se to. Chybová zpráva však nebude obsahovat nic o nesouhlasných verzích, nýbrž jen „nenalezený symbolický odkaz“.

I když je tento způsob označování verzí velmi důmyslný, může být někdy kontraproduktivní. Často se totiž stává, že genksyms vygeneruje různé transformované hodnoty pro parametry, které se v podstatě shodují.

Navíc, symbolické značení verzí ani není zárukou kompatibility. Je schopno zachytit pouze malou část možných druhů změn v definici funkce, jež mohou způsobit zpětnou nekompatibilitu. Jestli
že změna interpretace některého z parametrů podprogramem register_chrdev způsobí ztrátuzpětné kompatibility, přípona verze se nezmění – parametr má stále stejný typ. Neexistuje způsob, jak by to v programu insmod bylo možné nějakou volbou podobnou -f obejít.
V některých případech tedy není používání symbolického značení verzí rozumné.Pochopitelně, máte-li základní jádro přeložené se symbolickým značením verzí, musí být všechnyzaváděné moduly přeloženy stejně a naopak. Jinak máte zaručeno, že se neustále budete setká-vat s chybami „nenalezený symbolický odkaz“.

Provoz několika jader

Nyní, když vidíme, jak často se vyskytují verze zaváděných modulů odlišné od verzí základního jádra, vzniká otázka, co se systémem, který má několik verzí jádra systému (tj. že si jádro může-te vybírat při spouštění systému). Musíte mít jistotu, že při zavádění jádra A bude přidán modul zkompilovaný pro jádro A a naopak při zavádění jádra B bude přidán modul zkompilovaný pro jádro B.

Konkrétně to znamená, že při aktualizaci jádra by bylo velmi nerozumné se zbavovat starého
jádra, dokud nebudete mít jistotu, že nové jádro funguje. Obvykle k tomu využíváme rozlišovací schopnost programu modprobe, který se orientuje v orga-nizaci souborů zaváděných modulů, jak je popsána v kapitole „Kde jsou uloženy soubory zavá-děných modulů?“, a zavádí moduly z příslušných podadresářů v závislosti na tom, které jádro je v činnosti.

Hodnotu uname -release, která je rovna jménu podadresáře prohledávaného programem mod-probe, nastavíte v hlavním souboru makefile jádra při kompilaci jádra a nastavování proměnných VERSION, PATCHLEVEL, SUBLEVEL a EXTRAVERSION hned na začátku.

Symboly SMP

Je-li symbol definovaný nebo odkazovaný kódem vytvořeným pro symetrické víceprocesorové (symmetric multiprocessing, SMP) počítače, obsahuje prefix symbolické verze kromě kontrolního součtu zmíněného shora také „smp“. To znamená, že byl vytvořen pro použití v systému, který může mít víc než jednu základní jednotku (CPU). V konfiguračním procesu jádra Linuxu (make config atd.) můžete pomocí volby CONFIG_SMP zvolit, zda bude modul způsobilý i k činnosti na SMP počítači.

Používáte-li tedy symbolické značení verzí, nenalezené symboly se vyskytnou tehdy, je-li základní jádro způsobilé k činnosti na SMP a zaváděný modul není způsobilý nebo naopak.
Pokud symbolické značení verzí nepoužíváte, nic se neděje. Všimněte si, že obecně není důvod k tomu, abyste kompilovali jádro, které není způsobilé k čin-nosti na SMP, ani v případě, že máte jen jednu základní jednotku. Způsobilost není důvodem k tomu, abyste museli mít několik základních jednotek. Existují však počítače, na nichž nebude možné zavést jádro způsobilé k činnosti na počítači SMP, neboť dojde k závěru, že počet základ-ních jednotek je nula!

Když nejste oprávněni používat nějaký symbol

Majitelé copyrightu ke kódu jádra poskytují svoje programy veřejnosti na základě licence, která opravňuje veřejnost k pořizování a používání kopií, avšak s určitými omezeními. Licence může například stanovit, že kopii můžete volat pouze z programu, který

vzhledem k veřejnosti podléhá podobné licenci.
(Je to zmatené? Zde je příklad: Honza napíše zaváděný modul, který obsahuje podprogramy prokomprimaci dat, jež mohou využívat i jiné moduly. Svůj program poskytuje veřejnosti na základěGNU Public License (GPL). V souladu s některými interpretacemi tato licence říká, že pořídíte-li sikopii Honzova modulu, nesmíte dovolit Mařenčinu modulu, aby volal tyto komprimační podpro-gramy, pokud také neposkytne svůj zdrojový kód veřejnosti. Jde tedy o to, aby Mařenka zpří-stupnila svůj zdrojový kód.)

Z důvody podpory a prosazení takové licence může její držitel uzpůsobit svůj program tak, abyexportoval symboly pod speciálním jménem, které se skládá ze skutečného jména symbolu dopl-něného o prefix „GPLONLY“. Běžný zavaděč nebude schopen takové odkazy vyřešit. Příklad:Bobův modul poskytuje službu bobsService() a deklaruje ji prostřednictvím symbolu GPL. Tatoslužba je exportována pod jménem GPLONLY_bobsService. Odkazuje-li se Mařenčin modul naslužbu bobsService, běžný zavaděč ji nenajde a Mařenčin modul nezavede.

Moderní verze insmod však ví, že nenajde-li symbol bobsService, musí hledat symbolGPLONLY_bobsService. Neudělá to však, dokud Mařenčin modul nedeklaruje, že podléhá veřej-né licenci GPL.

Účelem tohoto omezení je zabránit jiné osobě, resp. programu, v náhodném porušení licence(nebo v porušení, jež je důvěryhodně označeno za náhodné). Je ovšem velmi snadné obejít tako-vé omezení úmyslně.

Narazíte-li na tuto chybu, pravděpodobně ještě používáte velmi starý zavaděč (insmode), který
nic neví o GPLONLY.Jedinou další možnou příčinou může být, že autor modulu napsal zdrojový kód, který nelze zavéstdo žádného jádra. V takovém případě autora není třeba nutit, aby kód zpřístupnil veřejnosti.

Zaváděný modul musí mít zavedeny spolupracující moduly

Stejným způsobem, jako musí být zaváděný modul kompatibilní s jádrem, musí být také kompa-tibilní s moduly, které využívá ke své činnosti (tj. volá jejich podprogramy). Z důvodu jednodu-chosti jsme se v předchozí části omezili jen na základní jádro.

O parametrech zaváděných modulů

Velice vhodné je porovnat parametry, které jsou předány do zaváděných modulů, s parametry pře-danými do modulů, jež jsou součástí základního jádra, a to proto, že zaváděné moduly mohou být provozovány obojím způsobem.

Výše jsme viděli, že parametry předáváte zaváděnému modulu příkazem insmod, v němž zadáte například io=0x300. Modulu (obvykle ovladači – příklady dále se týkají především nastavení hard-waru), který je součástí základního jádra, předáváte parametry prostřednictvím parametrů jádra. Obvykle zadáváte tyto parametry za název jádra ve výzvě programu lilo. Jinou možností je zadat parametry příkazem append v konfiguračním souboru programu lilo.
Jádro inicializuje zaváděný modul při zavádění a vestavěný modul při spouštění systému. Vzhledem k tomu, že existuje pouze jediný řetězec zaváděcích parametrů jádra, musí existovat způsob, jak identifikovat příslušnost k jednotlivým modulům. Pravidlo je takové, že k modulu jmé
nem xyz patří zaváděcí parametry jádra, které se také jmenují xyz. Hodnota zaváděcích paramet-rů jádra je libovolný řetězec, který může dávat smysl pouze tomuto modulu. Proto také můžete narazit na zaváděný modul, jehož jediným parametrem je jeho jméno. Napří-
klad ovladač Mitsumi CDROM zavedete příkazem:
insmod mcd mcd=0x340¨

Vypadá legračně, když máte parametr jménem mcd místo třeba io, avšak důvodem je konzisten-ce pro případ, že by mcd byl součástí základního jádra a zaváděcími parametry budou znaky mcd=0x340, jimiž zadáte V/V adresu portu.

Parametry jednotlivých modulů

V této kapitole si ukážeme, jak zjistit parametry konkrétního zaváděného modulu. Kde to bude možné, pomůžeme si odkazem na spolehlivější dokumentaci příslušného modulu (kterou nejspíše udržuje ten, kdo udržuje také kód modulu).

> Poznámka
>
> Původní kapitolu s popisem modulů jsme v knižním vydání vynechali kvůli neaktuálnosti. Popisovala moduly jádra staré řady 2.4 (a staré verze k tomu), zatímco většina dnešních systémů již používá jádro 2.6. Původní kapitolu z anglické verze dokumentu najdete na adrese *http://tldp.org/HOWTO/Module-HOWTO/individual.html.*

Příkaz modinfo

Základní informace o parametrech modulu rychle zjistíte například pomocí příkazu modinfo, minimálně se dozvíte uložení

modulu a seznam jeho parametrů. V lepších případech (ne každý modul toto má) obsahují informace získané programem modinfo i rozsah hodnot a význam jed-notlivých parametrů. Ukázka:

modinfo ehci-hcd filename: /lib/modules/2.6.17-13mdv/kernel/drivers/usb/host/ehci-hcd.ko.gz license: GPL author: David Brownell description: 10 Dec 2004 USB 2.0 'Enhanced' Host Controller (EHCI) Driver alias: pci:v*d*sv*sd*bc0Csc03i20* depends: usbcore vermagic: 2.6.17-13mdv SMP mod_unload 686 gcc-4.1 parm: log2_irq_thresh:log2 IRQ latency, 1-64 microframes (int) parm: park:park setting; 1-3 back-to-back async packets (uint)

Užitečné informace najdete též ve zdrojových kódech modulů. Jméno souboru se zdrojovým kódem bývá obvykle stejné nebo podobné jménu zaváděného modulu, takže při jeho hledání můžete použít například program locate:

locate ehci-hcd /lib/modules/2.6.17-13mdv/kernel/drivers/usb/host/ehci-hcd.ko.gz /usr/src/linux-2.6.17-13mdv/drivers/usb/host/ehci-hcd.c

Z výpisu programu je velmi jednoduché poznat, co je zkompilovaný modul a co jeho zdrojový kód v jazyce C.

Internetové zdroje

Mnohem aktuálnější informace naleznete na různých internetových zdrojích:

http://cs.wikibooks.org/wiki/Moduly_linuxov%C3%A9ho_j%C3%A1dra – odkaz na wiki knihu (v češtině), kde najdete popis jednotlivých jaderných modulů.
http://www.linux-faqs.org/ – další informace o jádře, jeho nastavení a modulech.

Technické podrobnosti

Jak pracují moduly

Program insmod vydá systémové volání init_module, aby zavedl modul do paměti jádra. Zave-dení modulu je snadné, ale jak jádro ví, že je má použít? Odpověď je taková, že systémové volá-ní init_module vyvolá inicializační program modulu ihned po jeho zavedení. Program insmod předá do init_module adresu podprogramu v zaváděném modulu, který má jméno init_modu-le jako jeho inicializační program.

(Je to trochu zamotané – každý zaváděný modul obsahuje podprogram (funkci) jménem init_module a základní jádro má systémové volání stejného jména, které je dostupné prostřed-nictvím podprogramu ze standardní knihovny C, který se také jmenuje init_module.)

Autor zaváděného modulu nastaví init_module tak, aby volal funkci jádra, která zaregistruje pod-programy obsažené v zaváděném modulu. Například podprogram init_module ovladače znako-vého zařízení může volat podprogram jádra register_chrdev, jenž společně s ostatními paramet-ry předá hlavní a vedlejší číslo zařízení, které bude ovládat, a adresu jeho vlastního „otvíracího" podprogramu. register_chrdev zaznamená v tabulkách základního jádra, že chce-li jádro otevřít toto konkrétní zařízení, musí nejdříve zvolat otevírací program v našem zaváděném modulu.

Avšak bystrý čtenář se nyní zeptá, jak může podprogram init_module zaváděného modulu znát adresu podprogramu register_chrdev základního jádra. To není systémové volání, nýbrž oby-čejný podprogram základního jádra. Jeho volání znamená předání řízení na jeho adresu. Jak tedy může náš zaváděný modul, který byl překládán v uctivé vzdálenosti od jádra, znát tuto adresu?

Klíčem k odpovědi je sestavení, které proběhne v době činnost insmod.Rozdíl mezi tím, jak toto sestavení proběhne v Linuxu 2.4 a v Linuxu 2.6, je zcela zásadní. Ve verzi
2.6 insmod pouze předá doslovný obsah souboru zaváděného modulu (soubor s příponou .ko) jádru a sestavení provede jádro. Ve verzi 2.4 provede sestavení insmod a předá jádru obraz plně sestaveného modulu připraveného k činnosti v paměti jádra. *Níže uvedený popis se vztahuje k jádru 2.4*.

Program insmod plní funkci sestavovacího/zaváděcího programu. Soubor zaváděného modulu obsahuje externí odkaz na symbol register_chrdev. insmod provede systémové volání query_module, aby našel adresy různých symbolů exportovaných stávajícím jádrem. Jedním z nich je i register_chrdev; query_module vrátí adresu register_chrdev a insmod jí nahra-dí odkaz na symbol register_chrdev v zaváděném modulu.

Máte-li zájem o informace, které insmod získává ze systémového volání query_module, poskyt-ne vám je obsah souboru /proc/ksyms. Poznamenejme ještě, že některé zaváděné moduly volají podprogramy v jiných modulech. Umož-ňují jim to části __ksymtab a .kstrtab v souborech zaváděných modulů. Obsahují seznam exter-ních symbolů v souborech zaváděných modulů, jež mohou být využívány budoucími zaváděnými moduly. Program insmod řekne jádru, aby přidalo symboly z __ksymtab a .kstrtab do tabul-ky exportovaných symbolů.

Vyzkoušet si to můžete tak, že vložíte zaváděný modul msdos.o a přesvědčíte se, že se objeví v souboru proc/ksyms symbol fat_add_cluster (což je jméno podprogramu v modulu fat.o). Každý následně vložený modul může předat řízení podprogramu fat_add_cluster, který ve sku-tečnosti běží v modulu msdos.o.

Sekce .modinfo

Soubor ELF (Executable and Linking Format) s přeloženým modulem obsahuje různě pojmenova-né sekce. Některé z nich jsou základními součástmi, např. sekce .text obsahuje spustitelný kódzaváděný zavaděčem. Můžete si vytvářet i vlastní sekce, které mohou využívat například speciál-ní programy. Jedna z takových sekcí se jmenuje .modinfo a je určena pro činnost zaváděnýchmodulů. Zaváděné moduly nemusí tuto sekci bezpodmínečně využívat, avšak je-li modul progra-mován pomocí maker, což se očekává, bude sekce .modinfo vytvořena, takže obecně lze říci, žeji moduly používají.
Rozdělení souboru na sekce (včetně .modinfo), pokud existují, můžeme prohlížet pomocí pro-gramu objdump. Například:Výpis všech sekcí souboru modulu msdos:

```
objdump msdos.o --section-headers
```

Výpis obsahu sekce .modinfo:

```
objdump msdos.o --full-contents --section=.modinfo
```

Interpretaci obsahu sekce .modinfo můžeme provést pomocí programu modinfo.Co tedy sekce .modinfo obsahuje a kdo tento obsah využívá? Využívá ho například program ins-mod:

- Obsahuje číslo verze jádra, pro které byl modul zkompilován, tj. toho zdrojového stromu

  jádra, jehož hlavičkový soubor byl použit pro překlad modulu.insmod tyto informace využívá, jak je vyloženo v kapitole „Nenalezené symboly“.
- Popisuje tvar parametrů doplňkového modulu. Program insmod využívá tyto informace k úpravě parametrů převzatých z příkazového řádku do tvaru potřebného pro nastavení počátečních hodnot datových struktur zaváděného modulu.

Sekce __ksymtab a .kstrtab

Dvě další sekce, které často najdete v souboru zaváděného modulu, se jmenují __ksymtab a .kstrtab. Dohromady tvoří seznam exportovaných symbolů dostupných z jiných částí jádra. Symbol je pouze textové jméno adresy v zaváděném modulu. Soubor modulu A se může odka-zovat na adresu v modulu B jménem (řekněme „getBinfo“). Když vložíte modul A později než modul B, program insmod může do modulu A vložit skutečnou adresu z modulu B, na niž byla data, resp. podprogram, getBinfo zaveden(a).

Další únavné podrobnosti o symbolickém adresování naleznete v kapitole „Jak pracují moduly“.

Symboly ksymoops

Program insmod přidává při zavádění do zaváděného modulu množinu exportovaných symbolů. Všechny tyto symboly jsou určeny k tomu, aby podporovaly program ksymoops, což je program, který interpretuje výpis chyby jádra, to je zpráva, kterou vypíše jádro, když detekuje vnitřní chybu ve svém těle (a následně ukončí proces). Zpráva obsahuje zejména množství hexadecimálních adres.

Program ksymoops tyto hexadecimální adresy prohlíží, vyhledává v tabulce symbolů (v Linuxu 2.4 jsou v /proc/ksyms, v Linuxu 2.6 jsou v /proc/kallsyms) a převádí je na symbolické adre-sy, které odpovídají symbolům v asemblerovském výpisu.Řekněme tedy, že máte zaváděný modul, který vám zhavaruje. Zpráva o chybě jádra obsahujeadresu instrukce, která způsobila chybu, a po programu ksymoops požadujete, aby vám řekl: 1)v kterém modulu je tato instrukce a 2) kde je umístěna v asemblerovském výpisu tohoto modu-lu. Podobné otázky vznikají i v souvislosti s adresami ve výpisu chyby jádra.
Aby na tyto otázky ksymoops dokázal odpovědět, musí znát zaváděcí adresy a délky různých sekcí modulu z tabulky symbolů jádra. Nuže, v Linuxu 2.4 insmod tyto adresy zná, takže pro ně pouze vytvoří symboly a přidá je k sym-bolům, které zavedl společně s modulem.

Konkrétně se tyto symboly jmenují (a naleznete je v souboru /proc/ksyms):

```
__insmod_name_Ssectionname_Llength
```

přičemž name je jméno modulu (tak jak je uvedeno v souboru /proc/modules), sectionname jejméno sekce, např. .text (nezapomeňte na úvodní tečku), length je dekadická délka sekce.
Hodnota symbolu je, pochopitelně, adresa sekce.Program insmod také přidá velice důležitý symbol, který říká, ze kterého souboru byl modulzaveden. Jméno tohoto symbolu je:

```
__insmod_name_Ofilespec_Mmtime_Vversion
```

přičemž name je jméno modulu jako ve výše uvedeném případě. filespec je specifikace souboru použitá k identifikaci souboru (na disku), který obsahoval zavá-děný modul při jeho zavádění. Poznamenejme, že nemusí mít stejné jméno a existuje několik spe-cifikací souboru, které se mohou odkazovat na tentýž soubor. Například ../dir1/mylkm.o a /lib/dir1/mylkm.o.
mtime je čas modifikace tohoto souboru ve standardním unixovém tvaru (vteřiny od r. 1969) hexa-

decimálně.version je verze jádra, pro niž byl modul zkompilován (stejná jako v sekci .modinfo). Je to hod-nota makra LINUX_VERSION_CODE v souboru linux/version.h. Například 132101.

Hodnota tohoto symbolu je bezvýznamná.

V Linuxu 2.6 to funguje jinak, ale zatím nevím jak.

Ostatní symboly

Další symboly, podobné symbolům ksymoops, přidá program insmod. Následující symbol říká, kde jsou v modulu uchovávaná data, která ukládá program rmmod.

__insmod_name_Plength

Symboly pro ladění

Další druh symbolů, které se vztahují k zaváděným modulům, jsou symboly kallsyms. Nejsou toexportované symboly a nejsou uloženy v souboru /proc/ksyms. Odkazují se na adresy v jádru,netýkají se ničeho jiného než modulu, v kterém se nacházejí, a neodkazuje se na ně nic kroměladicího programu. Jediným uživatelem symbolů kallsyms je kdb, což je ladicí program dodávanýs jádrem.

Ladicí aparát programu kallsyms je možno použít jak k ladění základního jádra, tak k ladění modu-lů. Se základním jádrem jej zkonfigurujete v době kompilace pomocí konfigurační volbyCONFIG_KALLSYMS. Když tak učiníte, jádro obsahuje symbol kallsyms pro všechny symboly v sou-boru základního jádra. Zda je jádro součástí ladicího aparátu kallsyms, zjistíte podle symbolu__start___kallsyms v souboru /proc/ksyms.

Do zaváděného modulu začleníte symboly kallsyms v době zavádění. Definici kallsyms zahrnetedo dat, která předáváte systémovému volání init_module při zavádění modulu. Program insmodto provede, když buď 1) zadáte volbu --kallsyms nebo 2) program insmod podle obsahu sou-boru /proc/ksyms zjistí, že jádro je součástí ladicího aparátu kallsyms. Všechny kallsyms defino-vané programem insmod jsou symboly v souboru zaváděného modulu. Zvídavým ještě prozra-dím, že to jsou symboly, s nimiž se setkali, když zadali příkaz nm na soubor s modulem.

Každý zavedený modul, který se účastní v kallsyms, má vlastní tabulku symbolů kallsyms. Kdyžse účastní i jádro, jednotlivé tabulky symbolů kallsyms jsou přidány do hlavní tabulky symbolůtak, aby ladicí program mohl vyhledávat symboly kdekoli v jádru. Jestliže se jádro neúčastní, ladi-cí program musí symboly pro určitý modul vyhledávat pouze v tomto modulu. Kdb to dělat neumí.Základní pravidlo tedy zní: Chcete-li ladit jádro, zadejte parametr CONFIG_KALLSYMS.

Poznamenejme, že sekce __kallsyms nemá nic společného se zaváděnými moduly, neboť je sou-částí modulu základního jádra. Základní jádro nemá žádnou výsadu vyšší úrovně při zavádění,takže musí mít zvláštní soubor, který umožní jeho účast v kallsyms.

Podobně ani program kallsyms nemá nic společného se zaváděnými moduly. Vytvoří jej sekce __kallsyms.Existuje i jiný druh ladicích symbolů – vytvoří jej gcc s volbou -g. Ke kallsyms nemají žádný vztah,nezavádějí se do paměti jádra a kdb je nepoužívá. Používá je však program kgdb (který si bereinformace jak z paměti jádra, tak i ze souboru modulu).

Přidělování paměti při zavádění

Tato kapitola je o způsobu přidělování paměti v Linuxu pro zavádění modulů. Není o dynamic
kém přidělování paměti moduly, které je stejné jako v kterékoli jiné části jádra. Paměť, v níž je zaváděný modul, je poněkud odlišná od paměti jádra. Základní jádro je vždy zave-deno do jedné velké souvislé oblasti reálné paměti, jejíž fyzická adresa je shodná s virtuální adre-sou. Je tomu tak proto, že základní jádro je vůbec to první ze systému, co je zavedeno do pamě-ti (jistěže kromě zavaděče) – a v tu dobu je k dispozici dostatek volné paměti. A protože jádro není možné stránkovat, zůstává trvale na svém původním místě.

V době zavádění modulu je však už celá paměť roztříštěná na kousky, takže nemůžete prostě zavést modul za jádro. Modul však musí být v souvislé části virtuální paměti (v adresovém pro-storu jádra), jež však pravděpodobně není fyzicky souvislá. Avšak *ani tuto paměť nelze stránko-vat*. Modul je tedy od začátku zaveden do několika fyzických úseků paměti a zůstává v nich, dokud není zrušen.

Některé základní jednotky využívají ke zrychlení přístupu do paměti určité vlastnosti základního jádra. Celé základní jádro může být například pokryto jednou položkou v tabulce stránek, a tudíž i jednou položkou ve stránkovém buferu (translation lookaside buffer, TLB). Tato položka TLB je přirozeně virtuálně vždy přítomná. Ke každému modulu je v tabulce jedna stránková položka odpovídající stránce, do níž je modul zaváděn. Když základní jednotka potřebuje některou strán-ku, příslušná položka mnohem častěji v TLB není, což znamená pomalejší přístup.

Účinek je pravděpodobně zřejmý.
Někdo tvrdí, že v Linuxu PowerPC je nevhodný způsob adresování příčinou neefektivního přecho-du mezi pamětí jádra a pamětí zaváděného modulu. Nemám o tom žádné spolehlivé informace. Značná část souvislé paměti základního jádra je vyhrazena opakovaně použitelné paměti – pamě-
ťová banka jádra. V některých verzích Linuxu se zavaděč modulů nejdříve pokusí získat pro zave-dení modulu paměť z této banky, a teprve když není k dispozici dostatečně velký úsek, použije k zavedení modulu virtuální paměť. Kód k tomuto mechanismu přidělování paměti pro Linux 2.5 poskytl v říjnu 2002 Andi Kleen. Tvrdí, že rozdíl je v rozmezí několika procent.

Vnitřní mechanismy

Zajímáte-li se podrobněji o vnitřní mechanismy činnosti jádra Linuxu vzhledem k zaváděným modulům, můžete považovat tuto kapitolu za úvod do problematiky. K vývoji, kompilaci a používání zaváděných modulů však další znalosti pravděpodobně nejsou nutné.

Kód pro zpracování modulů je ve zdrojových souborech kernel/module.c ve zdrojovém stromě Linuxu.Zavaděč modulů jádra (viz kapitola „Automatické zavádění a rušení modulů“) je v souboru ker-nel/kmod.c.

(Tak, moc toho pro začátek není, ale přinejmenším mám do budoucna odrazový můstek k další
mu psaní.)

Psaní vlastních zaváděných modulů

> Poznámka
>
> Velmi zajímavý materiál o programování jádra a jeho modulů najdete v seriálu „Vývoj jádra“, který připravuje Lukáš Jelínek pro časopis LinuxEXPRES, stránky viz *http://www.linuxexpres.cz.* Podobné informace najdete i ve starším, ale podobně obsáhlém seriálu Roberta Vojty na *http://www.linuxzone.cz/.*

Kompletní popis psaní vlastních zaváděných modulů naleznete v knize *The Linux Kernel Module Programming Guide* (*http://tldp.org/LDP/lkmpg*), autory jsou Peter J. Salzman, Michael Burian a Ori Pomerantz. Kniha vyšla i v tištěné podobě a existuje ve dvou verzích: pro Linux 2.4 a pro Linux

2.6. Ta první je mírně zastaralá a je v ní několik chyb. Následující popis obsahuje některé poznatky, které v tomto dokumentu ještě nebyly uvedeny, při-
nejmenším nikoli v ucelené podobě. Nejsou-li ve shodě se skutečností, sdělte to, prosím, autoro-vi tohoto návodu, aby v takovém případě mohl tuto kapitolu z dokumentu odstranit.

Jednodušší hello.c

Lkmpg je příklad nejjednoduššího zaváděného modulu na světě, hello-1.c. Mohl by však být ještě jednodušší a závisí to na tom, jak je v jádru nastaveno předávání zpráv o činnosti systému. Navíc program vyžaduje, aby v příkazu pro překlad byla zadána volba -D, neboť ve zdrojovém kódu nedefinuje určitá makra, ačkoli tam mají být.

Zde je zdokonalený nejjednodušší zaváděný modul na světě, hello.c.

```
/* hello.c *
* "Ahoj všichni" - verze zaváděného modulu jádra.
*

* Překládejte příkazem:
*

* gcc -c hello.c -Wall
*/


/* Uvete z hlavičkových souborů, jaký druh kódu chceme */ #define __KERNEL__ /* Jsme součástí jádra */
#define MODULE /* Nikoli permanentní část */

/* Standardní hlavičky modulů */ #include <linux/modversions.h> #include
<linux/module.h>
```

Přeložíme jednoduchým příkazem:

```
$ gcc -c -Wall -nostdinc -I /usr/src/linux/include hello.c
```

Volba -I předpokládá, že máte zdrojový kód, z něhož bylo základní jádro (základní jádro jádra, do kterého byste rádi zavedli hello.c) vygenerováno na obvyklém místě /usr/src/linux. Jste-li nato-lik masochističtí, že v jádru používáte symbolické značení verzí, zpracujte raději zdrojový kód jádra také příkazem make dep, který vytvoří soubory .ver, jež změní jména všech symbolů.

Poznamenejme nicméně, že často buď nejsou nainstalované hlavičky jádra vůbec nebo jsou chybné. Používáte-li jádro z distribučního CD, často z něho musíte hlavičky stáhnout zvlášť. Když už si hrajete s překladem zaváděných modulů, měli byste si přeložit i jádro, abyste přesně věděli, s čím
pracujete, a pak si můžete být absolutně jistí, že používáte odpovídající hlavičkové soubory. Volba -nostdinc není zcela nezbytná, je však dobrá, neboť vás může zbavit různých nepříjem-ností a také vám připomene, že jádro nemá k dispozici služby standardní knihovny C, které si možná v duchu spojujete s C samotným. Volba -nostdinc říká, že do vyhledávací cesty v hla-vičkových souborech nepatří „standardní“ adresáře. Zde jde zejména o adresář /usr/include.

Volba -c pouze říká, že chcete vytvořit pouze soubor .o, a to na rozdíl od implicitní volby v pro-gramu gcc, která vytvoří cílový soubor, sestaví jej s několika jinými standardními cílovými soubo-ry a vznikne tak něco, co lze volat z uživatelského procesu. Vzhledem k tomu, že tento modul nechcete volat, nýbrž jen přidat do jádra, fáze sestavování by neměla žádný smysl.

Volba -Wall (na jejímž základě kompilátor varuje před různými druhy pochybného kódu) obvyk-le není nutná, avšak tento program by neměl generovat žádná varování. V opačném případě je třeba odstranit příčinu.

Způsob kompilování jádra

Lkmpg (*http://tldp.org/LDP/lkmpg/*) obsahuje instrukce pro kompilaci zaváděného modulu (s tou výjimkou, že makro __KERNEL__ a obvykle také makro MODULE by měly být definovány ve zdro-jovém kódu, nikoli pomocí volby -D, jak navrhuje lkmpg). Za zmínku však stojí, že někteří pro-gramátoři jádra Linuxu jsou přesvědčeni, že jediným správným způsobem, jak vytvořit zaváděný modul, je přidat jej do kompletního zdrojového stromu a zkompilovat jej pomocí souborů make úplně stejně jako moduly, jež jsou součástí jádra.

Má to svoje výhody. Největší spočívá v tom, že je vyřešena situace, kdy programátoři změní způ
sob komunikace modulu se zbytkem jádra a v důsledku toho je nutno změnit i způsob kompilace. Na druhé straně z hlediska správy kódu pravděpodobně zjistíte, že je lepší mít vlastní kód oddě-lený od kódu Linuxu, a z hlediska programátorského zase, že překlad vlastního kódu musíte mít pod kontrolou, zejména když dojde ke změně.

Rubini a spol.: Linux Device Drivers

Nejoblíbenější knihou, která popisuje psaní ovladačů, je kniha *Linux Device Drivers*. Vydalo jinakladatelství O'Reilly's a napsali ji Alessandro Rubini, Jonathan Corbet a Greg Kroah-Hartman.
Tato kniha je vhodná i tehdy, když píšete jiný zaváděný modul než ovladač.První vydání popisuje Linux 2.0 s poznámkami k verzi 2.2. Druhé vydání (červen 2001) popisujeLinux 2.4. Třetí vydání popisuje Linux 2.6. Pochopitelně, kdo ví, jak to v Linuxu chodí, je mu jasné,že žádná kniha nemůže kvůli častým změnám popisovat výhradně určitou verzi. Distribuce Linu-xu 2.6 začala měsíc po zahájení prodeje třetího vydání této knihy a jsou v něm výrazné rozdílyoproti tomu, co popisuje kniha.

Druhé vydání této knihy je dostupné pod FDL (Free Documentation License). Lze si je přečíst naadrese http://www.xml.com/ldd/chapter/book/. Třetí vydání je dostupné pod licencí CreativeCommons Attribution-ShareAlike a najdete ho na http://lwn.net/Kernel/LDD3/.

V tištěné podobě dostanete tuto knihu v každém slušnějším knihkupectví. V Česku to bude horší,ale vyzkoušejte zásilkové knihkupectví *http://marecek.kup.to*, kde jsou schopni a ochotni podob-né knihy dovézt ze zahraničí. Podle neoficiálních informací chystá knihu o vývoji jádra také nakla-datelství Computer Press.

Čítač použití

Je zřejmé, že jádro se nepokouší odkazovat na kód modulu, který byl zrušen; tj. nesmíte odstra-nit modul v činnosti. Ovladač zařízení je v činnosti například tehdy, když je otevřen speciální sou-bor tohoto zařízení. Vzhledem k tomu, že existuje deskriptor otevřeného souboru, uživatel může zadat čtení z tohoto zařízení a provést je, přičemž jádro by mohlo chtít zavolat funkci v tomto ovladači. Je zřejmé, že po zrušení ovladače před tímto čtením by nastal problém – jádro by mohlo znovu chtít použít paměť, která obsahovala čtecí podprogram, a nikde není řečeno, na kterou instrukci předá řízení, když volá čtecí podprogram.

V původním návrhu modul zvyšuje a snižuje čítač volání o jedničku, aby správce poznal, zda jej může zrušit. Je-li to například ovladač souborového systému, přičte k čítači jedničku, když někdo připojí souborový systém daného typu, a při odpojení jedničku odečte.

Později bylo vytvořeno flexibilnější řešení. Modul zaregistruje funkci, kterou správce zavolá, když chce zjistit, zda může modul

zrušit. Vrátí-li funkce hodnotu true, znamená to, že je modul v čin-nosti a nelze jej zrušit. Vrátí-li hodnotu false, modul je nečinný a lze jej zrušit. Správce modulů před voláním této funkce modul uzamkne až do ukončení činnosti nebo přechodu do stavu spán-ku, a dokud neprovedete něco, co by mohlo znamenat, že modul nemůže být v činnosti mezi okamžikem, kdy ohlásíte „nečinný“ a ukončením činnosti.

Takže jak zaregistrujete tuto funkci, která říká, zda je modul v činnosti? Tak, že uložíte adresu do pole v deskriptoru modulu („struktura modulu“) bohužel nazvaného can_unload. Jméno je zvo-leno skutečně nešťastně, neboť booleovská hodnota, kterou vrací, je právě opačná, než co zna-mená „can unload“: Pravdivá, když správce modul *nemůže* zrušit.

Správce modulů se přesvědčí, že se nepokouší zrušit modul před tím, než se inicializační pod-program ukončil nebo přešel do stavu spánku, takže pole can_unload můžete v inicializačním podprogramu bezpečně nastavovat kdykoli kromě přechodu do stavu spánku.

Pole can_unload není příliš známé a používá se málokdy. Počínaje Linuxem 2.6 už neexistuje. Ať už používáte klasický čítač volání nebo pole can_unload, existují případy, kdy není jisté, zda modul nebyl zrušen v průběhu činnosti. Vytváří-li modul vlákno jádra, které provádí kód modu-lu, je téměř nemožné být si absolutně jist, že vlákno zmizelo před odstraněním modulu. Existují různé další služby jádra, kterým jste mohli předat adresy v těle modulu, přičemž tyto služby korektně neoznámí, že už předané adresy vymazaly. Dřív býval tento problém horší než dnes. Když modul například vytvořil v souborovém systému soubor pro nějaký proces, nemohli jste neodstranitelnost modulu nijak změnit po celou dobu, kdy některý proces zpracovával čtecí nebo zápisové podprogramy pro daný soubor. Tento problém i jeho jiné projevy byly odstraněny tak, že byl *vně* modulu vytvořen kód, který rozpoznal, že pou-žitá adresa může ukazovat do modulu, takže je nutno zvýšit nebo snížit čítač volání o jedničku. Kde je tato funkce implementovaná, bývá část struktury pojmenovaná „owner“, což je pomůcka modulu (tj. adresa struktury modulu).

Možná, že v budoucí verzi Linuxu budou tyto problémy odstraněny. Do té doby můžete tvůrcům jenom držet palce. Existuje názor, že tento typ problému je natolik složitý, že by bylo dobré modu-ly nerušit. Počínaje Linuxem 2.6 lze pomocí konfigurační volby jádra CONFIG_MODULE_UNLOAD rušení modulů zakázat.

Rozdíly mezi verzemi Linuxu

Za zmínku v této kapitole stojí množství verzí Linuxu, jež koluje po světě. Na rozdíl od proprie-tárního softwaru, kdy jedna společnost bedlivě střeží jméno a vytváří jen malý počet přesně defi-

novaných verzí, různé Linuxy vyvíjí množství nezávislých osob a všechny verze se nazývají Linux. Základní verze řídí Linus

Torvalds a jako hlavní verze je distribuuje kernel.org. Jsou jedinými ver

zemi, které oprávněně mohou být nazývány „Linux 2.4“, „Linux 2.6.6“ atd. Správu starších řad jádra (2.4, 2.2, 2.0) předává Linux jiným vývojářům. Jenomže málokdo je skutečně používá – pouze z nich vycházejí následné modifikace, které jsou nesprávně označovány stejně jako původní verze: např. Linux 2.6.6. Tvůrci by k nim měli přidá-vat alespoň pomlčku a nějakou příponu. Často používané verze Linuxu firmy Red Hat bohužel jako příponu používají pouze číslo, např. Linux 2.6.6-12. (Bylo by například vhodné, kdyby pří-pona nějak charakterizovala firmu, dejme tomu Linux 2.6.6-rh12).

Připomínám, že „Linux“ v tomto dokumentu znamená jádro; pokud bychom hovořili o operačním systému GNU/Linux, situace by byla ještě mnohem složitější.

Linux 2.4 – Linux 2.6 Sestavování modulů v jádru

Největší změna, která nastala v zaváděných modulech mezi verzemi Linux 2.4 a 2.6, má interní charakter: Byl změněn způsob zavádění těchto modulů. Kromě změny přípony zaváděných modu-lů však pravděpodobně nezaznamenáte žádné další rozdíly, neboť pro jejich správu se používají nástroje vyšší úrovně, jejichž rozhraní zůstalo beze změny.

Před verzí 2.6 interpretoval program v uživatelském prostoru soubor typu ELF (.o), sám jej sesta-vil pro běžící jádro a sám vygeneroval jeho konečný binární obraz, který pak předal jádru. Jádro jej pak pouze uložilo do paměti a provedlo některé další úkony. Ve verzi 2.6 jádro sestavuje modul samo a uživatelský program pouze předá jádru obsah souboru ELF s modulem. Z toho důvodu musí soubor obsahovat některé další informace a kvůli identifikaci těchto souborů byla přípona „o“ nahrazena příponou „ko“ („kernel object“). Například ovladač sériového zařízení, který se v Linuxu 2.4 jmenoval serial.o, se v Linuxu 2.6 jmenuje serial.ko. Místy narazíte i na kom-primovanou podobu, tedy serial.ko.gz.

Z toho důvodu byl pro Linux 2.6 vytvořen zcela nový balík. Ve srovnání s rozsáhlým sestavova

cím programem ve verzi 2.4 je v tomto balíku program insmod triviálním programem. Na druhé straně je postup při vytváření zaváděných modulů poněkud složitější. Soubor typu .ko je nutno vytvořit ze souboru .o, který zpracujete pomocí programu modpost (je součástí dodáv-ky zdrojové verze Linuxu); modpost vytvoří zdrojový program v C, který popisuje sekce potřeb-né pro soubor typu .ko. Takový soubor nazýváme soubor .mod, neboť je zvykem přidávat do jména souboru znaky „.mod“.

Soubor .mod přeložíte a soubor .ko vytvoříte tak, že výsledek překladu sestavíte s původním sou-

borem .o.Soubor .ko obsahuje jméno, které dostane doplňkový modul po zavedení. Toto jméno nastavítepři překladu souboru .mod pomocí volby -D, která vytvoří makro KBUILD_MODNAME.

Tato změna některé věci rozhodně komplikuje – například zadávání jména, které dostane zavá-děný modul při zavádění. V Linuxu 2.4 bylo toto jméno jedním ze vstupů jádra. Jméno zavádě-ného modulu vytvořil program insmod a předal je jádru, přičemž ho bylo možno zadat explicit-ně pomocí volby -o. Ve verzi 2.6 už takové volání systému ani volba -o v programu insmod nee-xistují. Jméno je součástí souboru ELF (souboru typu .o), který předáváte jádru. Jméno je jehosoučástí i v případě, když je implicitní, a chcete-li je změnit, musíte je změnit v souboru ELF a pakteprve předat programu insmod.

Neexistence funkce „modul v činnosti“

V Linuxu 2.6 bylo pole can_unload (viz kapitola „Čítač použití“) zrušeno.

CONFIG_MODULE_UNLOAD

Generování jádra můžete zkonfigurovat tak, že vytvoříte jádro, v němž je zcela zakázáno rušení zaváděných modulů, což lze vzhledem k problémům s rušením modulů v činnosti považovat za klasický úkrok stranou. Viz kapitola „Čítač použití“.

Čítač odkazů

Rozhraní, které kód zaváděného modulu používá k práci s čítačem odkazů, bylo přemístěno.

Copyright v souvislosti se zaváděnými moduly

Trvalkou mezi dotazy je dotaz, zda zaváděný modul spadá pod licenci GPL, když je pod ní distri-buováno jádro. Je například možné dodávat zaváděné moduly pouze v binárním tvaru? V tétokapitole si popíšeme problémy kolem copyrightu k zaváděným modulům, které jsou stejně zají-mavé jako složité. Říkám na rovinu: Nedobereme se k žádnému závěru. Právo v této otázce ještěnení ustálené a existuje celá řada pádných, avšak protichůdných argumentů.

Připomeňme, že zákony o copyrightu jsou v různých zemích různé. Pokud byste v jedné zemidokázali správně odpovědět na některou otázku, v jiné zemi může být vše jinak. Ve skutečnostivšak jsou zákony v různých zemích dosti podobné a nepůjdeme do takových podrobností, aby-chom mezi nimi hledali rozdíly.
Pusťme se tedy do toho. Začněme tím, co je to copyright. Copyright je především právo jednotlivce zabránit jiným lidem v kopírování něčeho. Je to zákon-né právo, nikoli morální. To znamená, že má přinést nějaký praktický užitek a netýká se něčeho, o čem si lidé myslí, že je přirozené. Copyright zejména usiluje o to, aby autor dostal zaplaceno za to, co vytvořil. Někdo je přesvěd-čen, že to je ten pravý účel, protože autor má přirozené právo na hodnotu, kterou vytvořil. Avšak z historického hlediska to není skutečný účel copyrightu. Zaplatit autorovi za práci je až druhot-
ným účelem, autor především tvoří. Autor bude spíše trávit čas psaním a investovat svoje peníze,když za to dostane zaplaceno.Existence zákona o copyrightu splňuje tento účel pouze přibližně. Všichni známe případy, kdy byl zákon použit k převodu jmění způsobem, který v žádném případě nepřispívá k tomuto účelu.Hudební vydavatel například zakáže kopírování písničky, za kterou nijak nezaplatil. Zákonodárceto nazývá nepřímou škodou – jak zákon o copyrightu na jedné straně podporuje tvorbu, v jinýchpřípadech vytváří zcela nesmyslná omezení.

Když však vezmeme zaváděné moduly, v zákoně o copyrightu existuje jiná, mnohem složitějšíoblast, na niž se vztahuje: odvozené dílo. Copyright dává autorovi právo zakázat druhé osoběvytvářet odvozené dílo, což ovšem *není* kopie ničeho, co by autor napsal. Takže co to vlastně je?

Definice odvozeného díla není jednoduchá, zkusme si nejdříve uvést několik příkladů: Když pře-ložíte knihu z angličtiny do francouzštiny, francouzská verze je dílo odvozené od anglického díla.Když k románu připíšete novou kapitolu, je to dílo odvozené od románu. Když napíšete úplněnovou knihu o Harrym Potterovi se stejnými postavami a reáliemi, je to dílo odvozené od všechknih o Harrym Potterovi. Když namalujete Dilbertovu postavičku na blahopřání k narozeninám, jeto dílo odvozené od všech knih a seriálů o Dilbertovi.

A tak se pomalu dostáváme k zaváděným modulům. Mnoho lidí je přesvědčeno, že zaváděný modul je dílo odvozené od jádra Linuxu. Nadace Free Software Foundation řekla ano, je to tak. Linus Torvalds řekl, že je to tak jen někdy. Žádný soud ještě nerozhodl ani jedním, ani druhým způsobem.

Nyní se tedy vraťme ke copyrightu na Linux. Pro účely této diskuse, když řekneme „Linux“, bude-me tím myslet obsah archivu (komprimovaného např. programem tar), který stáhnete z kernel.org jako regulérní jádro Linuxu.

Kdo k němu vlastní copyright? Takových lidí je mnoho. Téměř každý, kdo přispěl nějakým kódem do Linuxu, si vyhradil svůj copyright. Za některé z těchto příspěvků byla vyplacena mzda, copy-right tedy náleží zaměstnavateli autora. Nejznámějším majitelem copyrightu je Linus Torvalds, ačkoli má copyright jen k velmi malé části Linuxu.

Jak tito lidé uplatňují svoje právo? Určitě nikdo s každým z nich neuzavírá smlouvu, to by bylo nepraktické. Všichni však nabízejí veřejnosti (stejnou) licenci. Tato licence je zdokumentována a poskytována jako součást balíku a je známa pod názvem General Public License (GPL). Není specialitou Linuxu. Byla vytvořena nadací Free Software Foundation ještě před tím, než vznikl

Linux, a Linus ji zvolil, když byl jediným majitelem copyrightu. A Linus do balíku nezařadí žádný kód, dokud jej autor nenabídne pod stejnou licencí.

Dalo by se říci, že licence je v tomto případě stav, kdy vlastník copyrightu povolí něco, co má právo zakázat, např. pořizování kopií nebo vytváření odvozeného díla. To, že je licence veřejná, znamená, že se vztahuje na veřejnost, nikoli na určitou osobu, která je vlastníkovi copyrightu známa.

GPL umožňuje dělat s kódem prakticky cokoli – téměř jako kdyby se vlastník copyrightu zřekl všech práv, které mu z copyrightu plynou. Ne však zcela: Některé nitky zůstaly nepřestřižené. Licence stanoví určité podmínky, a chcete-li povolení, musíte je splnit. Podmínky, o nichž zde hovoříme, spočívají v jedné ze dvou věcí, případně v obou, v závislosti na tom, jak čtete licenci (tento dokument není jednoznačný). Buď a) pokud distribuujete dílo odvozené od Linuxu, musí-te poskytnout zdrojový kód k celému odvozenému dílu; nebo b) pokud distribuujete dílo obsa-hující Linux, musíte poskytnout kód k celému dílu.

Nyní se tedy dostáváme ke skutečné otázce: Jaké máte právo na distribuci zaváděného modulu jádra, který napíšete? Mnozí se pochopitelně domnívají, že můžete zaváděné moduly distribuovat bez omezení, včetně pouze binární podoby, a že majitelé copyrightu k Linuxu nemohou nic namítat. Nedistribuujete nic jiného než svůj vlastní kód, který není žádným odvozeným dílem. Mnozí to skutečně takto dělají a jsou přesvědčeni, že právě toto je hlavním přínosem zaváděných modulů.

Existuje však i jiný názor. I když je zaváděný modul původní dílo, je dílem odvozeným od Linu-xu a vlastníci copyrightu k Linuxu mají právo kontrolovat jejich distribuci. Jediné povolení, které od nich můžete získat, je GPL a tato licence ukládá, že musíte poskytovat i zdrojový kód. Další informace najdete v par. 2 v licenci GPL, v českém překladu například na adrese *http://staff.cesnet.cz/~lhotka/gnugpl-cz.html*.

Podívejme se na problém zaváděného modulu jako díla odvozeného z Linuxu poněkud blíže. Argument pro spočívá v tom, že psaní zaváděného modulu je jako psaní další kapitoly k románu, což, jak víme, je odvozené dílo. Zaváděný modul je jako kapitola, neboť je určen výlučně k tomu, aby byl součástí celého jádra Linuxu a v jiném kontextu je nepoužitelný. V okamžiku použití je pevně svázán se zbytkem Linuxu. Skutečnost, že k jeho překladu obvykle používáte hlavičkové soubory Linuxu, je důkaz, že jde jen o rozšíření Linuxu (jde ovšem o jiný argument, než že pří-kaz `#include` ve skutečnosti znamená, že tento hlavičkový soubor distribuujete jako součást vaše-ho souboru s modulem). A také je nutno uvést, že pokud zvolíte tento způsob, zaváděný modul zavedený v průběhu výpočtu je pevně spojen s jádrem Linuxu. Je-li psaní modulu do linuxového stromu odvozeným dílem, musí jím být i psaní zaváděného modulu. Je-li aktualizováno jádro, často je nutno provést také aktualizaci zaváděného modulu.

Společnost Wasabi Systems, která prodává jádra s volnější licencí, než má Linux, vydala dokument (http://www.wasabisystems.com/gpl/lkm.html), v němž dokazuje mnohem podrobněji, že zaváděný modul je dílo odvozené z Linuxu, a z toho důvodu byste měli zveřejňování zaváděné-ho modulu pouze v binární podobě důkladně zvážit.

Argument proti říká, že zaváděný modul je něco, co na sebe s Linuxem pouze vzájemně působí, nikoli co je součástí Linuxu. Přirovnává zaváděný modul k uživatelskému programu, který komu-nikuje s jádrem prostřednictvím systémových volání, nebo ke klientskému programu FTP (který není dílem odvozeným od programu FTP na serveru).

Problém také zasahuje do oblasti, kdy kniha a počítačový program nejsou analogiemi, neboť počí-tačový program něco dělá, zatímco kniha pouze člověku zprostředkovává myšlenky. Zavádění modulu se více podobá nasazování nástavce k vysavači než vkládání stránek do knihy. A víme, že návod k nástavci není dílem odvozeným od návodu k vysavači.

Tak, nic dalšího už k tomu nemohu dodat. Zdá se, že to nejspíše bude otázka míry a dělicí čáru musí někudy vést soud.

Pokud byste ovšem chtěli učinit praktické rozhodnutí, vězte, že po mnoho let existovaly velice oblí-bené zaváděné moduly (nejznámějším příkladem jsou asi ovladače síťových adaptérů Nvidia) pouze v binární podobě, a nikdo nikdy nikoho nežaloval za porušení copyrightu. A Linus Torvalds vli-vem jiných důvodů než právních řekl, že zaváděné moduly v binární podobě mu plně vyhovují.

Co říci o symbolech GPL-ONLY? Projektanti jádra vybrali některé symboly, které se používají pouze v rozhraních mezi zaváděnými moduly a jádrem jako výlučně GPL. Řetězec „GPL-ONLY“ je součástí jejich jména, což činí tento úmysl zcela zřejmým, a je to pokyn pro linuxový zavaděč modulů, aby je po vás požadoval, což s sebou přináší povinnost zahrnout do zaváděného modulu určitý kód, kterým stvrzujete, že váš modul bude podléhat předpokládané licenci GPL.

To ovšem pravděpodobně nemá žádný právní význam. Není-li zaváděný modul dílem odvozeným od Linuxu, nemají projektanti jádra k dispozici žádný způsob, jak zablokovat vložení anotace GPL do kódu a distribuovat modul v libovolné binární podobě. Pokud zaváděný modul *je* dílem odvo-zeným od Linuxu, nepřítomnost klasifikace symbolu „pouze GPL“ pravděpodobně není dostateč-ná k tomu, aby dávala právo užívat jej jako zaváděný modul pouze v binární podobě. Text licen-ce se o tom nezmiňuje. Přinejlepším můžete symboly „pouze GPL“ považovat jenom za džentl-menský slib, že vás nikdo nezažaluje za používání všech ostatních symbolů v zaváděném modu-lu, který je pouze v binární podobě.

Další informace

V poslední kapitole uvedu ještě několik drobností, které se nevešly nikam jinam.

Zavádění systému bez ovladače disku

Ve většině systémů musí být ovladač ATA disku součástí základního jádra, neboť kořenový systém souborů je na ATA disku[1] a bez ovladače jej jádro nemůže připojit (méně často potřebuje číst

[1] Tento typ disku pravděpodobně znáte pod označením „IDE“. Přesně řečeno, IDE je nesprávný název. IDE je zkratka technologie „Integrated Drive Electronics“, kterou používají všechny moderní diskové mechaniky, zejména pak všechny mechaniky SCSI. První běžně používané mechaniky IDE byly mechaniky ATA a názvy se poněkud zaměnily. ATA, podobně jako SCSI, je přesná specifikace elektronických signálů, příkazů atd.

z kořenového souborového systému i zaváděný modul). Jestli však přece jen chcete, aby ovladač pro čtení kořenového souborového systému byl zaváděný modul, zde je návod, jak toho lze dosáhnout pomocí tzv. *initrd*:

„Initrd“ je v Linuxu zkrácený pojem pro „initial ramdisk“. Jeho prostřednictvím zavaděč (pravdě-podobně LILO nebo Grub) zavádí souborový systém do paměti (jako ramdisk) ještě před spuště-ním jádra. Po spuštění jádro požádá o připojení ramdisku jako kořenového souborového systému. Ovladač disků skutečného kořenového systému souborů a veškerého dalšího softwaru, který je nutno zavést, dáte do souborového systému ramdisk. Nakonec spustíte startovací programy (které jsou uloženy na ramdisku) tak, aby připojily skutečný (diskový) souborový systém jako kořeno-vý. Všimněte si, že ramdisk nepotřebuje ke své činnosti ovladač.

To vás ovšem nezbavuje nutnosti mít v základním jádru Linuxu: 1) ovladač souborového systému na ramdisku a 2) spustitelný interpret programů na ramdisku.

Uchovávaná data

Některé zaváděné moduly jsou nastaveny tak, že si mezi jednotlivými zavedeními uchovávají urči-té informace. Nazýváme je uchovávaná data. Když takový modul rušíte programem rmmod, tento program vezme určité hodnoty z pracovní paměti zaváděného modulu a uloží je do souboru. Při příštím zavádění je vrátí na původní místo v modulu.

Viz volba --persist v příkazech insmod a rmmod. Tato možnost existuje v

Linuxu od listopadu 2000.

Dokumenty související

K modulům, které jsou součástí Linuxu (tj. jsou šířeny se základním jádrem), můžete někdy nalézt dokumentaci i v podadresáři Documentation zdrojového kódu Linuxu. Mnohé zaváděné moduly mohou být připojeny k jádru alternativně. V takovém případě jim bude-te předávat parametry prostřednictvím „příkazového řádku“ jádra, čímž je míněn prompt v průbě-hu zavádění. Podrobnosti naleznete v návodu *The BootPrompt HOWTO* od Paula Gortmakera (Paul.Gortmaker@anu.edu.au) v původní verzi Linux Documentation Project na adrese (http://www.tldp.org/).

Nezapomeňte, že jako poslední a nejdůvěryhodnější instance dokumentace vždy poslouží zdrojo-vé kódy Linuxu a příslušného zaváděného modulu.

Softwarový RAID

Úvod

Tento text zahrnuje pouze „nový“ RAID přítomný v jádrech řady 2.4 a 2.6. Nepopisuje „starý“ RAID, který je součástí jádra řady 2.0 a 2.2. Zaměřuje se na konkrétní verzi vrstvy pro softwarový RAID, jmenovitě verzi 0.90 napsanou Ingo Molnarem a dalšími. Jedná se o standardní RAID vrst-vu v linuxovém jádře verze 2.4 a 2.6 a v některých distribucích i jádra verze 2.2. Podpora softwa-rového RAID 0.90 je k dispozici ve formě dodatečných oprav (patches) i pro jádra 2.0 a 2.2 a je vesměs považována za mnohem stabilnější než původní podpora RAID v těchto jádrech.

Domácí stránky návodu jsou na adrese *http://unthought.net/Software-RAID.HOWTO/* a jsou prů-běžně aktualizovány. Návod původně napsal Jakob Ostergaard a byl poskládaný z e-mailů mezi ním, Ingo Molnarem (*mingo@chiara.csoma.elte.hu*), jedním z vývojářů RAID a několika dalšími lidmi v e-mailové konferenci (l*inux-raid@vger.kernel.org*) o linuxovém RAID. Na první verzi návo-du se podílel Emilio Bueso (*bueso@vives.org*).

Pokud máte v plánu používat nový RAID s jádry 2.0 nebo 2.2, budete muset nejdříve použít úpra-vy zdrojových kódů jádra, které jsou k dispozici na adrese *http://people.redhat.com/mingo.* Tato jádra nemají přímou podporu níže popsaného nového typu RAID. Proto je zapotřebí zmíněných oprav. Starší podpora RAID v jádrech 2.0 a 2.2 obsahuje mnoho chyb a postrádá několik důležitých vlastností, které má nový RAID.

Informace z následujících odstavců se vám mohou zdát velmi jednoduché, pokud již RAID trochu znáte, takže je můžete s klidem přeskočit.

Prohlášení

Povinné prohlášení: Všechny informace v textu jsou prezentovány „tak jak jsou“, bez jakýchkoli záruk. Pokud přijdete o všechna svá data, přijdete o svou práci, budete sražení nákladákem nebo cokoli jiného, není to ani moje vina, ani vývojářů. Mějte na paměti, že budete používat RAID a čer-pat informace z tohoto textu na vlastní riziko. Neexistuje žádná záruka, že software nebo násle-dující informace jsou na 100 % přesné. Před jakýmikoli pokusy si pečlivě zálohujte svá data. Je lepší mít svá data zabezpečená než později litovat.

Co je to RAID?

V roce 1987 publikovala Kalifornská univerzita v Berkeley článek o využití redundantních polí lev-ných disků (A Case for Redundant Arrays of Inexpensive Disks (RAID) – *http://www-2.cs.cmu.edu/~garth/RAIDpaper/Patterson88.pdf*). V článku je popsáno několik typů diskových polí označovaných zkratkou RAID. Základní myšlenka RAID je ve spojení několika malých nezá-vislých disků do jednoho pole, které svým výkonem předčí jeden velký drahý disk (SLED – Sing-le Large Expensive Drive). A navíc se toto pole jeví systému jako jedna logická jednotka nebo disk.

Čas mezi výpadky diskového pole (MTBF – Mean Time Between Failure) pak bude stejný jako čas pro samostatný disk podělený počtem disků v poli. Taková životnost pole ale může být pro náro-ky aplikací příliš malá. Nicméně disková pole mohou být vytvořena jako odolná proti výpadku jednoho disku uložením dat redundantně na více disků najednou.

V článku bylo definováno 5 typů architektury diskových polí (RAID-1 až RAID-5). Všechny jsou odolné proti výpadku disku a každý nabízí jiné výhody ve vlastnostech a výkonnosti. Jako dopl-něk k těmto pěti architekturám se stalo zvykem nazývat neredundantní pole disků jako RAID-0.

V dnešní době jsou některé původní úrovně RAID (hlavně 2 a 3) používány pouze ve velmi spe-cializovaných systémech (a nejsou ani podporovány ovladači pro linuxový softwarový RAID). Oproti tomu přibyla „lineární“ úroveň a také RAID-0 se často kombinuje s RAID-1.

Terminologie

V následujícím textu bude „RAID“ znamenat „linuxový softwarový RAID“. Návod nepopisuje žádné vlastnosti hardwarového RAID. Navíc neobsahuje ani žádné popisy softwarového řešení pro RAID na jiných operačních systémech.

Pokud je popisováno nastavení RAID, tak je vždy vhodné odkazovat na čísla disků a jejich veli-kosti. Ve všech případech bude písmeno N (Number) použito jako počet aktivních disků v poli (nepočítaje volný disk). Pokud nebude řečeno jinak, pak písmeno S (Size) bude velikost nejmen-šího disku v poli. Písmeno P (Performance) označuje rychlost disku v poli v MB/s. Ačkoli to nemu-sí být v reálných podmínkách pravda, předpokládá se, že všechny disky jsou rychlostně na stejné úrovni.

Slova „zařízení“ a „disk“ mají v textu shodný význam. Zařízení použité pro sestavení RAID jsou obvykle jen diskové oddíly, ne nutně celé disky. Nicméně protože spojení diskových oddílů z jed-noho disku nemá většinou význam, jsou slova zařízení a disk myšlena jako „diskové oddíly na růz-ných fyzických discích“.

Úrovně RAID

Níže je uvedený stručný popis úrovní RAID podporovaných linuxovými ovladači. Některé infor-mace tvoří jen základní přehled, ale přidal jsem i několik poznámek o specialitách linuxové imple-mentace v jednotlivých úrovních. Jestliže RAID znáte, můžete tuto kapitolu v klidu přeskočit.

Současné linuxové ovladače pro RAID podporují následující úrovně:

- Lineární mód

Dva nebo více disků jsou spojeny do jednoho fyzického zařízení. Disky jsou „spojeny“ za sebe, takže zápis na RAID probíhá lineárně. Nejdříve se zapisuje na disk 0 a po jeho zaplnění na disk 1 a tak dále. Disky nemusí mít stejnou velikost. Zde opravdu platí, že na velikosti nezáleží.
V této úrovni není žádná redundance. Jestliže jeden z disků havaruje, pak pravděpo-dobně přijdete o všechna svá data. Když ale budete mít štěstí, můžete nějaká data obnovit, protože na souborovém systému bude chybět jen určitá souvislá oblast.
Výkonnost čtení a zápisu se pro samostatné operace nijak nezvýší. Ale zrychlení může být v určitých případech patrné a to ve chvíli, kdy jeden uživatel přistupuje k soubo-rům uloženým na prvním disku a další uživatel k datům na druhém.

- RAID-0

Také nazývaný jako „stripe“ mód (prokládání). Zařízení by měla (ale nemusí) mít stejnou velikost. Operace na diskovém poli budou rozděleny mezi zařízení. Například zápis většího souboru může být rozdělen po 4kB blocích, takže jsou zapsány 4 kB na disk 0, pak 4 kB na disk 1, pak 4 kB na disk 2 a poté znovu 4 kB na disk 0 a tak stále dokola. Pokud je jeden z disků v poli větší než ostatní, pak je jeho místo sice využito, ale při větším zaplnění pole RAID budou probíhat zápisy jen na tento disk, což se samozřejmě projeví na výkonnosti.
Stejně jako u lineárního režimu zde není žádná redundance. Na rozdíl od lineárního režimu ale v případě výpadku přijdete o všechna data. Pokud odstraníte jeden disk z pole typu RAID-0, nebude v poli chybět souvislý blok dat, ale po celém zařízení bude mnoho malých prázdných děr. Nástroj e2fsck nebo jiné nástroje pro obnovu souboro-vého systému z takového zařízení pravděpodobně příliš neobnoví.
Operace čtení a zápis se výrazně zrychlí, protože probíhají paralelně z několika disků. Obvykle je výkonnost hlavním důvodem pro provozování RAID-0. Pokud jsou řadiče dostatečně rychlé, může se výkonnost velmi přiblížit N*P MB/s.

- RAID-1

Jedná se o první mód se skutečnou redundancí. RAID-1 se dá použít na dvou a více discích s žádným nebo více volnými (záložními) disky. RAID-1 udržuje přesnou kopii informací (zrcadlí je) z jednoho disku i na ostatních v poli. Disky musí mít samozřejmě stejnou velikost. Pokud je jeden větší než ostatní, bude výsledný RAID velký maximálně jako nejmenší disk.
Žádná data nejsou dotčena, dokud není odebráno (nebo rozbito) více než N-1 disků. Pokud jsou k dispozici záložní disky a systém přežije havárii disku (což se pravděpodobně nestane například při chybě SCSI ovladačů nebo pádu IDE řadiče), pak je okamžitě po zjištění problému zahájena rekonstrukce zrcadleného disku na záložní.
Rychlost zápisu je obvykle horší než v případě samostatného disku, protože identická kopie dat je zapisována na každý disk v poli. S velkými diskovými poli úrovně RAID--1 to může být opravdu problém, protože jednotlivými kopiemi dat se zahltí PCI sběr-nice. Jde o jednu z mála oblastí, kde má hardwarový RAID navrch oproti softwarové-mu, protože zapisovaná data pak neprochází PCI sběrnicí, ale přímo hardwarovým řadičem. Rychlost čtení je dobrá, obzvláště je-li v systému více operací pro čtení nebo operací náročných na vyhledávání dat. Ovladač RAID obsahuje velmi dobrý algoritmus pro určení disku, jehož pozice je nejblíže požadovaným datům. Protože pohyb hlav (hledání pozice) je na současných discích časově velmi náročný (doba vyhledání činí 6 ms, což odpovídá přečtení 123 KB při rychlosti 20 MB/sec), je určení správného disku pro čtení dat velmi znatelné na výkonnosti celého pole.

- RAID-4
  - Jedná se o nepříliš používanou úroveň. Lze ji použít pouze pro tři a více disků. Na roz-díl od zrcadlení disků udržuje na jednom z nich pouze paritní bity a na zbylé disky zapisuje podobně jako na RAID-0. Protože jeden z disků je vyhrazen pro paritní infor-mace, je velikost pole (N-1)*S, kde S je velikost nejmenšího disku z pole. Stejně jako v RAID-1 by měly mít disky stejnou velikost, nebo se prostě budete muset smířit s veli-kostí pole podle nejmenšího disku.

Při pádu jednoho disku může být paritní informace použita pro rekonstrukci všech dat. Při pádu dvou disků jsou všechna data ztracena.
Důvodem, proč se RAID-4 příliš nepoužívá, je, že paritní informace je uložena na jediném disku. Proto musí být paritní informace aktualizována při každé změně dat na zbylých discích. Paritní disk se tedy stává úzkým hrdlem pro výkonnost, pokud není výrazně rychlejší než ostatní disky. Pokud ale máte mnoho pomalých disků a jeden znatelně rychlejší, může pro vás být tato úroveň velmi užitečná.

- RAID-5

Pravděpodobně nejpoužívanější RAID v případě, když máte k dispozici velké množství fyzických disků a chcete mít nějakou redundanci. RAID-5 je možné využít u třech a více disků s žádným nebo více záložními. Výsledná velikost pole bude stejná jako u RAID--4, tedy (N-1)*S. Velký rozdíl je ale v tom, že paritní informace je rovnoměrně rozklá-dána mezi všechny disky a zabraňuje se tak problému úzkého hrdla jako u RAID-4.
Pokud selže jeden z disků, tak díky paritní informaci zůstanou uložená data v pořád-ku. V případě použití záložního disku opět začne po zjištění problému okamžitá rekon-strukce nefunkčního disku na volný. Všechna data jsou ale ztracena, jestliže selžou disky dva nebo více.
Operace čtení a zápis jsou obvykle rychlejší, ale je velmi složité předpovědět jak moc. Čtení je stejně rychlé jako u RAID-0, zápis může být náročnější (pro správnou kalkula-ci paritní informace je nejdříve nutné přečíst data z ostatních disků před vlastním zápi-sem) nebo stejný jako u RAID-1. Rychlost zápisu hodně závisí na množství operační paměti systému a způsobu používání diskového pole. Hodně rozptýlené zápisy jsou obvykle výrazně pomalejší.

Požadavky

Návod předpokládá použití linuxového jádra 2.4 nebo vyššího. Nicméně, jak již bylo zmíněno, je možné zprovoznit softwarový RAID i na upravených verzích 2.2.x nebo 2.0. Úpravy jádra a nástro-je jsou k dispozici na *http://people.redhat.com/mingo.* Jádro, jeho úpravy a balík raidtools by měly ve verzích co nejvíce odpovídat. Nejsou-li k dispozici odpovídající verze úprav jádra, může být někdy lepší použít starší jádro.

Pokud používáte novější distribuci GNU/Linuxu založenou na jádře 2.4 a vyšším, tak máte prav-děpodobně k dispozici i korespondující verzi balíku raidtools.

Proč RAID?

Existuje mnoho dobrých důvodů, proč RAID používat. Patří mezi ně například spojení několika fyzických disků do jednoho většího „virtuálního“ zařízení, zvýšení rychlosti nebo redundance. Je ale velmi důležité si uvědomit, že úkolem RAID není nahradit správné zálohování. Některé druhy diskových polí sice pomohou zamezit ztrátě dat při výpadku jednoho disku, ale RAID vám nepomůže obnovit omylem smazaná data (například příkazem rm -rf /). RAID vám také nepo-může zachovat data, jestliže server s diskovým polem potká nějaká nestandardní událost (zloději, záplavy, zemětřesení, invaze Marťanů apod.)
Obecně vám redundantní diskové pole pomůže udržet systém v běhu i v případě častého hard-warového problému (výpadku jednoho disku). Ale nejedná se o kompletní řešení zabezpečení vašich dat. Je velmi důležité si tento fakt uvědomovat.

Zařízení a podporované souborové systémy

Linuxový RAID je schopen pracovat na většině blokových zařízení. Nezáleží na tom, jestli se jedná o IDE nebo SCSI disky nebo jejich kombinaci. Někteří lidé také používají, s menším či větším úspěchem, síťové blokové zařízení (NBD – Network Block Device). Od té doby, co je zařízení linuxového RAID samo blokovým zařízením, je v podstatě možné vytvá-řet diskové pole nad jinými RAID zařízeními. To například umožňuje používat RAID-10 (RAID-0 na několika RAID-1 zařízeních) jednoduchým spojením funkcionality RAID-0 a RAID-1. Jsou ale podporovány i jiné více exotické konfigurace. Například RAID-5 nad RAID-5.

Vrstva RAID nemá vůbec nic společného s vrstvou souborového systému. To v podstatě znamená, že můžete na diskové pole umístit jakýkoli souborový systém stejně jako na jiné blokové zařízení.

Výkonnost

RAID je často nasazován jako řešení výkonnostních problémů. Ačkoli RAID opravdu může být řešení, které hledáte, neznamená to, že zákonitě vaše problémy vyřeší. Může být mnoho jiných příčin výkonnostních problémů a diskové pole je řešením pouze pro některé z nich (viz první kapitola s výkonnostní charakteristikou jednotlivých úrovní).

Swap na RAID

Není důvod používat RAID pro oddíl odkládací paměti (swap) kvůli zvýšení výkonnosti. Samotné jádro podporuje rozložené swapování na několik zařízení. Stačí dát zařízením stejnou prioritu v souboru /etc/fstab.

Ukázka hezkého souboru /etc/fstab:

```
/dev/sda2 swap swap defaults,pri=1 0 0 /dev/sdb2 swap swap defaults,pri=1 0 0 /dev/sdc2 swap swap defaults,pri=1
0 0 /dev/sdd2 swap swap defaults,pri=1 0 0 /dev/sde2 swap swap defaults,pri=1 0 0 /dev/sdf2 swap swap
defaults,pri=1 0 0 /dev/sdg2 swap swap defaults,pri=1 0 0
```

Výše uvedené nastavení provádí swapování paralelně na sedm SCSI zařízení. Jádro má v sobě tuto

vlastnost obsaženou již velmi dlouho, takže žádný RAID není potřeba.
Jiným důvodem pro používání RAID na oddíl swap může být vysoká dostupnost. Například pokud nastavíte systém na startování ze zařízení typu RAID-1, pak systém pravděpodobně vydrží výpa dek jednoho disku. Systém se ale určitě zhroutí, pokud má swap oddíl na poškozeném disku. Umístění oddílu swap také na RAID-1 zmíněný problém řeší.

Samozřejmě se vede mnoho diskusí ohledně stability oddílu swap na zařízení RAID. Je to alenekončící debata, která značně závisí i na jiných vlastnostech jádra. V době psaní textu se totořešení zdálo perfektně stabilní. Stejně byste ale měli provést několik zátěžových testů systému,dokud nebudete s jeho stabilitou plně spokojeni.
Můžete také vytvořit soubor s odkládací pamětí na souborovém systému na zařízení typu RAID nebo vytvořit swap přímo na zařízení RAID. Jakkoli se vám zlíbí. Jak již bylo zmíněno, RAID je jen blokové zařízení.

Proč mdadm?

Klasické raidtools jsou standardním balíkem pro ovládání linuxových RAID zařízení, takže použí-vání programu mdadm není nutné.

Poznámka

V současné době je tomu již právě naopak a mdadm se stává dominantním pro ovládání diskových polí. V některých distribucích již ani nenajdete raidtools ve standardně dostup ných balících a k dispozici je jen mdadm.

Pokud se vám zdají raidtools těžkopádné a nedostatečné, může pro vás být mdadm (Multiple Devices ADMin) extrémně užitečný. Může být používán jako náhrada raidtools nebo jako jeho doplněk.

Nástroj mdadm napsal Neil Brown (*http://www.cse.unsw.edu.au/~neilb/),* systémový inženýr z aus-tralské University of New South Wales a vývojář jádra. Nástroj je momentálně ve verzi 2.5 a uká-zalo se, že je velmi stabilní. Všeobecně je v diskusních fórech přijímán více než kladně a je prav-děpodobné, že se v budoucnu více rozšíří.

Hlavní rozdíly mezi mdadm a raidtools jsou:

mdadm umí diagnostikovat, monitorovat a shromažďovat detailní informace o stavu diskových polí
mdadm je jednoduchý centrální program a ne kolekce roztroušených programů, což znamená společnou syntaxi příkazu pro správu diskového pole RAID
mdadm umí provádět většinu operací bez konfiguračního souboru a implicitně žádný nepoužívá
pokud mdadm nějaký konfigurační soubor potřebuje, pak také pomůže s vytvořením jeho obsahu

Zařízení

Zařízení softwarového RAID jsou takzvaná „bloková“ zařízení, stejně jako kterýkoli normální disk nebo diskový oddíl. Zařízení typu RAID je „sestaveno“ z několika jiných blokových zařízení. Napří-klad RAID-1 může být sestaven ze dvou obyčejných disků nebo ze dvou diskových oddílů (na různých fyzických discích – viz popis RAID-1).

Žádné další požadavky na sestavení zařízení typu RAID nejsou. Znamená to hodně velkou volnost při plánování vašeho řešení pro RAID. Například můžete vytvořit RAID zařízení z kombinace IDE a SCSI disků, stejně jako můžete sestavit RAID nad jiným zařízením typu RAID (užitečné hlavně pro RAID-10, kdy jednoduše vytvoříte dvě disková pole typu RAID-1 z normálních disků a nakonec vytvoříte diskové pole typu RAID-0 nad těmito zařízeními).

V následujícím textu bude používáno slovo „zařízení“ ve významu „disk“, „oddíl“ nebo „zařízení typu RAID“. „Zařízení“ jednoduše představuje „linuxové blokové zařízení“. Může se jednat o cokoli od SCSI disku až po síťové blokové zařízení. Častokrát budou tato „zařízení“ jednoduše nazývána „disky“, protože jimi většinou ve skutečnosti jsou.
Nicméně zařízení mohou být v diskovém poli v různých stavech. Může se jednat například
o „záložní disk“, o „poškozený disk“ nebo to může být normální plně funkční zařízení aktivně pra

cující v diskovém poli.Níže jsou popsány dva speciální stavy zařízení, jmenovitě „záložní disk“ a „poškozený disk“.

Záložní disk

Záložní disk je takový, který není v diskovém poli aktivní, dokud jiný disk neselže. Jakmile je dete-kováno selhání jednoho disku, je tento poškozený disk označený jako nefunkční a okamžitě začí-ná rekonstrukce dat na první volný záložní disk.

Takže záložní disky přidávají příjemné extra zabezpečení, hlavně pro RAID-5, ke kterým je často obtížný fyzický přístup. Můžete si tedy dovolit nechat běžet systém chvíli i s poškozeným diskem, protože stále máte zajištěnou redundanci dat.

Přesto si nemůžete být ani s tímto diskem jisti, že systém korektně poběží po pádu jednoho disku. Vrstva RAID se sice umí dobře postarat o výpadek disku, ale může být poškozený například ovladač SCSI disku nebo se může zaseknout IDE řadič nebo cokoli jiného.

Mimo jiné, jakmile se nastartuje rekonstrukce na záložní disk, RAID ovladač začne číst ze všech zbývajících disků, aby mohl vytvořit redundantní informace. Jestliže se na více discích během času vytvořily vadné sektory, může samotná rekonstrukce vyvolat poškození „dobrého“ disku. To samo-zřejmě vede ke kompletnímu výpadku pole. Jestliže pravidelně provádíte zálohování celého sou-borového systému na diskové pole, pak je velmi nepravděpodobné, že byste se do takové situace dostali. O důvod více, proč pravidelně zálohovat. Znovu připomínám, RAID není náhrada záloh.

Poškozený disk

Pokud ovladač RAID zvládne bez problémů výpadek disku, tak jej označí jako vadný a jeho rekon

strukce okamžitě začne na první volný záložní disk.
Poškozený disk je stále vidět a je součástí pole. Ovladač se k němu ale chová jako k neaktivní
části souborového systému.

Hardwarové problémy

Následující text popisuje některá hardwarová omezení, která mohou nastat při provozování soft warového RAID. Jestliže stavíte RAID kvůli vysokému výkonu, měli byste zajistit, aby diskové řadiče byly dostateč-ně rychlé. Neměli byste používat čtrnáct UW-SCSI disků na jednom UW řadiči, když každý disk zvládne 20 MB/s, zatímco řadič má propustnost „pouze“ 160 MB/s. Stejně tak byste měli mít pouze jeden disk na jednom IDE řadiči. Provoz dvou disků na jednom řadiči v režimu master/slave má značné dopady na výkonnost. IDE zařízení jsou opravdu pomalá, pokud přistupují k více než jednomu disku na řadiči. Všechny novější základní desky mají samozřejmě alespoň dva IDE řadiče, takže můžete použít dva disky pro diskové pole, aniž byste museli kupovat novou desku. Přídavné IDE řadiče jsou dnes ale opravdu levné, takže je snadné a dostupné použít i 6–8 IDE disků.

Konfigurace IDE

Je opravdu možné postavit RAID nad IDE disky, a to dokonce s velmi dobrým výkonem. Ve sku-tečnosti ceny dnešních IDE disků a řadičů dělají z těchto disků záležitost, kterou byste se měli určitě zabývat, pokud plánujete nové diskové pole typu RAID.
Fyzická stabilita: IDE disky jsou tradičně vyráběny jako mechanicky slabší než SCSI disky. Záruka těchto disků je typicky 2 roky na rozdíl od SCSI zařízení, která mívají záruku 3–5 let. Nebylo by spravedlivé říkat, že IDE disky jsou automaticky horší již z výroby. Ale je třeba si uvědomit, že IDE disk od jednoho výrobce má vyšší pravděpodobnost výpadku než podobný SCSI disk téže značky. Nicméně jednotlivé typy jsou mechanicky úplně stej-né jak pro IDE, tak pro SCSI disky. Sečteno a podtrženo: Každý disk se dříve nebo pozdě-ji pokazí a měli byste na to být připraveni.
Integrita dat: Dřívější IDE zařízení neměla žádné zajištění, že odeslaná data z řadiče budou opravdu stejným způsobem zapsána na disk, což bylo způsobené absencí parity, kontrolních součtů apod. Od standardu Ultra-DMA ale IDE disky již počítají kontrolní sou-čty na příchozích datech, takže je velmi nepravděpodobné, že by se provedl chybný zápis. Každopádně sběrnice PCI nepočítá ani paritu, ani kontrolní součty a je využívána jak pro IDE, tak pro SCSI disky.
Výkonnost: O výkonnosti se příliš rozepisovat nebudu. Zde je opravdu krátký popis:

IDE disky jsou rychlé, i když neexistují verze (v době psaní textu), které mají 10 000 nebo 15 000 otáček za minutu jako jejich SCSI protějšky (disky o 10 000 otáčkách jsou již dostupné i v IDE provedení).
IDE generují větší zatížení procesoru než SCSI (je to však opravdu důležité?).
Používejte pouze *jeden* disk na jednom řadiči, jinak se výkon výrazně sníží.

- Zotavení se z pádu: Ovladač IDE zařízení obvykle vydrží pád poškozeného disku. Ovla-dač RAID označí disk jako poškozený, a pokud používáte RAID typu 1 nebo vyšší, tak by systém měl bez problémů pracovat, dokud nebudete mít čas jej odstavit kvůli údržbě (výměně disku).

Je *velmi* důležité, abyste používali pouze *jeden* IDE disk na jedné IDE sběrnici. Nejenže dva disky mohou snížit výkonnost, ale pád jednoho disku obvykle způsobí chybu na sběrnici a tím pádem přestanou fungovat všechny disky na dané sběrnici. V konfiguracích RAID, kde je dovolen výpa-dek disku (úrovně 1, 4, 5), je možné chybu na jednom disku zvládnout, ale výpadek dvou (těch dvou, které jsou nedostupné, kvůli pádu sběrnice, na které je vadný jeden disk) znemožní nor-mální fungování pole. Stejně tak pokud se poškodí disk s nastavením „master“, tak buď disk s nastavením „slave“ nebo řadič budou nepříjemně zmateni aktuální konfigurací. Jedna sběrnice, jeden disk. To je základní pravidlo pro disky IDE.

Existuje mnoho levných IDE řadičů pro PCI sběrnice. Můžete je pořídit obvykle se dvěma nebo čtyřmi sběrnicemi za cenu okolo 500 Kč. Pokud uvážíte, o kolik levnější jsou disky IDE oproti dis-kům SCSI, vychází diskové pole z IDE disků jako velmi dobré řešení. Pokud jste však ochotni smí-řit se s relativně malým počtem disků (obvykle okolo osmi), které můžete zapojit do běžných systémů.

Rozhraní IDE má ve větších polích zásadní problém s kabely. I kdybyste měli dostatek PCI slotů, je velmi nepravděpodobné, že budete schopni vměstnat do počítače více než 8 disků, aniž by docházelo k poškození dat způsobených příliš dlouhými kabely.

Hot Swap (výměna disku za běhu)

Ačkoli výměna disků za běhu je do určité míry podporována, stále to není nic jednoduchého.

Výměna za běhu – disky IDE

Nedělejte to! Rozhraní IDE vůbec nepodporuje výměnu disků za běhu systému. Samozřejmě že to může za určitých okolností fungovat. Například pokud máte ovladač pro IDE zkompilován jako modul a znovu jej nahrajete po výměně disku. Stejně tak ale můžete skončit s poškozeným řadi-čem a budete čelit mnohem delší odstávce systému, než kdybyste jej regulérně shodili a disk vyměnili.

Hlavní problém (kromě komplikací s elektrickým proudem, kdy si můžete zničit hardware) je, že sběrnice IDE musí znovu skenovat disky po jejich výměně. Zatímco novější linuxová jádra pod-porují znovu skenování sběrnice (za pomoci nástroje hdparm), nové načtení diskových oddílů je stále problematické. Jestliže je nový disk na 100 % stejný jako starý (geometrie apod.),

tak se výměna může podařit, ale vydáváte se opravdu na tenký led.

Poznámka

V dnešní době jsou již hodně rozšířené IDE disky s rozhraním SATA (Serial ATA), které na rozdíl od rozhraní ATA v určitých konfiguracích výměnu disku za běhu systému podporu-jí. Neznamená to ovšem, že můžete jen tak vytáhnout kabel z disku a věřit, že se nic nesta-ne. Vždy se jedná o řešení, kdy musíte vlastnit speciální výměnné rámečky a řadiče, které výměnu za běhu přímo podporují. V některých případech dokonce i výrobci disků uvádě-jí, jestli je daný typ disku odolný pro výměnu za chodu. Pokud tedy budete pečlivě vybírat své komponenty, můžete získat konfiguraci, ve které je „hot swap“ řešitelný i pro disky IDE.

Výměna za běhu – disky SCSI

Klasická zařízení SCSI také nejsou určena pro výměnu za běhu systému. Nicméně to *může* fun-govat. Jestliže váš ovladač SCSI podporuje znovu načtení sběrnice a odpojování/připojování zaří-zení, můžete být schopni disky vyměnit za běhu. Každopádně na normální sběrnici SCSI pravdě-podobně nemůžete odpojit zařízení, zatímco je stále v provozu. Ale znovu říkám, je možné, že to bude fungovat (a je také možné, že skončíte se spáleným hardwarem).

Vrstva SCSI *by měla* vydržet pád disku, ale ne všechny ovladače jsou schopny výpadek zvládnout. Jestliže váš ovladač SCSI odejde spolu s diskem, pak odejde celý systém a výměna disků za běhu přestává být zajímavá.

Výměna za běhu – disky s SCA

Výměna disků s SCA je za běhu možná. Bohužel to ale není tak jednoduché, jak by mělo, je to však možné a bezpečné.Výměna RAID zařízení, diskového zařízení a nastavení správných parametrů host/channel/id/lun(řadič/kanál/SCSI ID/logické číslo jednotky) vypadá například následovně:

- Uložte tabulku oddílů (partition table) disku (pokud je stále čitelná):
  sfdisk -d /dev/sdb > partitions.sdb
- Odeberte vyměňovaný disk z pole:
  raidhotremove /dev/md0 /dev/sdb1
- Získejte parametry Host, Channel, ID a LUN vyměňovaného disku pohledem do:
  /proc/scsi/scsi

Odeberte disk ze sběrnice:
Ověřte, že disk byl korektně odebrán pohledem do:

echo “scsi remove-single-device 0 0 2 0” > /proc/scsi/scsi /proc/scsi/scsi

Odpojte disk ze SCA panelu a vložte nový disk:
Připojte nový disk ke sběrnici:

echo “scsi add-single-device 0 0 2 0” > /proc/scsi/scsi

(toto by mělo roztočit disk)

- Vytvořte novou tabulku oddílů na disku podle tabulky, kterou jste si dříve uložili:
  sfdisk /dev/sdb < partitions.sdb
- Přiřaďte disk zpět do pole:
  raidhotadd /dev/md0 /dev/sdb2

Argumenty pro příkaz „scsi remove-single-device“ jsou: Host, Channel, Id a Lun. Jejich čísla může te najít v souboru /proc/scsi/scsi.Výše uvedený postup byl vyzkoušen a prověřen na systému s disky SCA od IBM na řadiči Adaptec SCSI. Pokud narazíte na problémy nebo přijdete na snazší způsob, prosím, dejte o tom vědětdo diskusní skupiny o linuxovém RAID.

Konfigurace pro RAID

Obecná instalace

Pro konfiguraci jakékoli úrovně RAID budete potřebovat následující:

Jádro. Přednostně jádro řady 2.6 nebo 2.4, případně jádro řady 2.0 nebo 2.2 s příslušnými úpravami.
Ovládací nástroje (*RAID tools*).
Trpělivost, pizzu a váš oblíbený kofeinový nápoj.

Ovládací nástroje jsou standardně obsaženy ve většině dnešních linuxových distribucí.Jestliže máte systém s podporou pro RAID, měli byste mít k dispozici soubor /proc/mdstat.Pamatujte si jej, tento soubor je váš přítel. Pokud zmíněný soubor nemáte, je pravděpodobné, ževaše jádro RAID nepodporuje. Prohlédněte si, co soubor obsahuje, příkazem cat /proc/mdstat.
Mělo by v něm být, že máte nastavenou správnou RAID „osobnost“ (resp. úroveň RAID) a že
žádné zařízení RAID není momentálně aktivní.
Vytvořte oddíly, které chcete přidat do zařízení typu RAID.

Stažení a instalace mdadm

Aktuální verzi programu mdadm můžete získat na adrese http://www.cse.unsw.edu.au/ ~neilb/source/mdadm/. Pro kompilaci a instalaci programu mdadm, jeho dokumentace, manu-álových stránek a příkladů proveďte typické make install.

tar xvf ./mdadm-2.5.3.tgz cd mdadm-2.5.3 make install

Jestliže používáte distribuci založenou na balících RPM, můžete stáhnout i hotový balíček na adre-se
http://www.cse.unsw.edu.au/~neilb/source/mdadm/RPM.
rpm -ihv mdadm-2.5.3-1.i386.rpm

Pro Debian Woody (3.0) nebo pozdější stačí nainstalovat balík příkazem:

apt-get install mdadm

Gentoo má balíček dostupný ve svém repozitáři balíčků. Můžete použít:

emerge mdadm

Distribuce mohou mít také již připravený balík – prohledejte repozitáře vaší distribuce. Nyní se pusťme do konkrétních příkladů.

Lineární úroveň

Takže máte 2 nebo více oddílů, které nemusí mít stejnou velikost (ale samozřejmě mohou) a které

chcete spojit k sobě.Nastavte soubor /etc/raidtab, který bude popisovat vaši konfiguraci. Já jsem nastavil raidtabpro dva disky a soubor vypadal následovně:

raiddev /dev/md0 raid-level linear nr-raid-disks 2 chunk-size 32 persistent-superblock 1 device /dev/sdb6 raid-disk 0 device /dev/sdc5 raid-disk 1

Záložní disky nejsou v této konfiguraci podporovány. Jestliže jeden z disků selže, pak selže celé
pole. Lineární úroveň nemá žádné informace, které by na záložní disk zapisovala. Pravděpodobně přemýšlíte, proč jsem definoval *chunk-size* (velikost bloku), když lineární úroveň jen spojuje disky do jednoho velkého pole bez jakéhokoli paralelismu. Samozřejmě máte pravdu, je to divné. Prostě vložte pro chunk-size nějakou hodnotu a dál se tím nemusíte zabývat.
Pole vytvoříte příkazem:

mkraid /dev/md0

Příkaz vytvoří vaše pole, zapíše perzistentní superbloky a pole spustí.Pokud používáte program mdadm, pak vám bude stačit jediný příkaz pro vytvoření pole:

mdadm --create --verbose /dev/md0 --level=linear --raid-devices=2 /dev/sdb6 /dev/sdc5

Parametry hovoří samy za sebe. Výstup příkazu by pak měl vypadat následovně:

mdadm: chunk size defaults to 64K mdadm: array /dev/md0 started.

Zkontrolujte soubor /proc/mdstat. Měli byste vidět, že je pole aktivní.V tuto chvíli již můžete vytvořit souborový systém stejným způsobem, jako byste to dělali pro libo-volné zařízení. Stejně tak je můžete posléze připojit, vložit do souboru /etc/fstab a podobně.

RAID-0

Máte dvě nebo více zařízení přibližně stejné velikosti a chcete zkombinovat jejich úložnou kapa

citu a zároveň zvýšit jejich výkonnost paralelním přístupem.Nastavte soubor /etc/raidtab, který bude popisovat vaši konfiguraci. Soubor může napříkladobsahovat:

raiddev /dev/md0 raid-level 0 nr-raid-disks 2 persistent-superblock 1 chunk-size 4 device /dev/sdb6 raid-disk 0 device /dev/sdc5 raid-disk 1

Stejně jako u lineární úrovně nejsou podporovány záložní disky. RAID-0 nemá žádnou redundan

ci, takže při výpadku jednoho disku bude nedostupné celé pole.
Opět spusťte:

```
mkraid /dev/md0
```

pro vytvoření pole. Měly by se vytvořit superbloky a spustit raid zařízení. Podívejte se do soubo-ru /proc/mdstat, abyste viděli, co se stalo. Měli byste vidět, že je vaše zařízení v provozu. Zařízení /dev/md0 je nyní připravené pro naformátování, připojení, použití a zneužití.

RAID-1

Máte dvě zařízení přibližně stejné velikosti a chcete, aby se vzájemně zrcadlila mezi sebou. Pří-padně máte další zařízení, které chcete mít připravené jako záložní, a chcete, aby se stalo auto-maticky součástí pole ve chvíli, kdy jeden z aktivních disků selže.

Nastavte následovně soubor /etc/raidtab:

```
raiddev /dev/md0 raid-level 1 nr-raid-disks 2 nr-spare-disks 0 persistent-superblock 1 device /dev/sdb6 raid-disk 0 device /dev/sdc5 raid-disk 1
```

Pokud máte zmíněný záložní disk, přidejte jej na konec definice zařízení pomocí těchto řádků:

```
device /dev/sdd5
spare-disk 0
```

Nezapomeňte nastavit odpovídajícím způsobem parametr nr-spare-disks, který vyjadřuje počet záložních disků.
Momentálně je tedy vše připravené pro sestavení pole. Zrcadlené disky se musí spojit, respektive obsah (i když nepodstatný, protože zařízení zatím není naformátované) na obou zařízeních musí být synchronizován.

Spusťte

```
mkraid /dev/md0
```

pro zahájení synchronizace.
Opět zkontrolujte soubor /proc/mdstat. Měl by obsahovat záznam o tom, že zařízení /dev/md0 je v provozu, že započala rekonstrukce dat a odhadovaný čas do jejího ukončení. Rekonstrukce probíhá v době klidu ostatních vstupně-výstupních operací. To znamená, že systém
by měl normálně reagovat, ačkoli informační diody disků budou intenzivně svítit.
Proces rekonstrukce je navenek transparentní, což pro vás znamená, že můžete zařízení již používat, i když rekonstrukce ještě probíhá. Zkuste během průběhu rekonstrukce disků zařízení naformátovat. Mělo by to fungovat. Stejně tak můžete během vytváření pole zařízení připojit a používat. Samozřejmě budete mít smůlu, pokud se v tomto okamžiku poškodí ten nesprávný disk.

RAID-4

> Poznámka
>
> Tuto konfiguraci jsem osobně netestoval. Níže uvedené nastavení je můj nejlepší odhad a ne něco, co mám někde v provozu. Jestliže používáte RAID-4, prosím, napište o tom autorovi (*jakob@unthought.net*) a podělte se o své zkušenosti.

Máte tedy tři nebo více zařízení skoro stejné velikosti a jedno zařízení je znatelně rychlejší než ostatní. Chcete je zapojit do jednoho většího pole a udržovat nějaké redundantní informace. Případně máte ještě nějaké množství volných disků použitelných jako záložní.

Vytvořte následovně obsah souboru /etc/raidtab:

```
raiddev /dev/md0 raid-level 4 nr-raid-disks 4 nr-spare-disks 0 persistent-superblock 1 chunk-size 32 device /dev/sdb1 raid-disk 0 device /dev/sdc1
        raid-disk 1 device /dev/sdd1 raid-disk 2 device /dev/sde1 raid-disk 3
```

Záložní disky přidejte stejným způsobem dle zvyklostí RAID (opět musíte posléze upravit parametr nr-spare-disks):

```
device /dev/sdf1
spare-disk 0
```

Pro vytvoření pole použijte stejný příkaz jako v předchozích případech:

```
mkraid /dev/md0
```

Před samotným formátováním zařízení byste si měli nejdříve přečíst kapitolu se speciálními nasta-veními pro příkaz mke2fs.

RAID-5

Máte tři nebo více zařízení přibližně stejné velikosti, které chcete spojit do jednoho většího pole, ale zároveň udržovat určitý stupeň redundance pro bezpečnost vašich dat. Případně máte i dané množství volných disků připravených jako záložních pro případ, že se některý z aktivních disků poškodí.

Pokud použijete N disků, kde velikost nejmenšího je S, bude výsledná velikost pole určena vzor-cem (N-1)*S. „Chybějící“ kapacita je určena pro paritní (redundantní) informace. Z toho důvodu pole zůstane funkční i v případě výpadku jakéhokoli disku. O data ovšem přijdete, pokud výpa-dek postihne dva a více disků.

Soubor /etc/raidtab nastavte následovně:

```
raiddev /dev/md0
        raid-level 5
        nr-raid-disks 7
        nr-spare-disks 0
        persistent-superblock 1
        parity-algorithm left-symmetric
        chunk-size 32
        device /dev/sda3
        raid-disk 0
        device /dev/sdb1
        raid-disk 1
        device /dev/sdc1
        raid-disk 2
        device /dev/sdd1
        raid-disk 3
        device /dev/sde1
        raid-disk 4
        device /dev/sdf1
        raid-disk 5
        device /dev/sdg1
        raid-disk 6
```

Záložní disky budou k poli přidány stejným způsobem jako v předchozích případech, například:

```
        device                                  /dev/sdh1
        spare-disk                              0
```

Velikost jednoho bloku 32 kB (chunk size) je optimální výchozí hodnota pro většinu použití sou-borových systémů této velikosti. Pole, na které je výše uvedený raidtab použitý, je velké 7 krát 6 GB = 36 GB (vzpomeňte si na (n-1)*s = (7-1)*6 = 36). Obsahuje souborový systém *ext2* s bloky
o velikosti 4 kB. Můžete použít větší hodnoty obou parametrů (chunk-size a velikost bloku sou-borového systému), jestliže máte mnohem větší kapacitu disků nebo na pole ukládáte velmi velké soubory.

Nechme povídání. Máte-li nastavený soubor /etc/raidtab, můžete otestovat, že vše opravdu fun-guje. Spusťte příkaz:

```
mkraid /dev/md0
```

a zkontrolujte, co se stalo. Při troše štěstí vaše disky začaly pracovat jako šílené ve chvíli, kdy zača
ly rekonstruovat pole. Pro aktuální informace se podívejte do souboru /proc/mdstat.Jestliže se podařilo zařízení úspěšně vytvořit, pak okamžitě započala rekonstrukce pole. Pole neníkonzistentní, dokud rekonstrukce neskončí. Avšak pole je opět plně funkční (samozřejmě kroměpřípadu havárie disku) a můžete jej naformátovat a používat i během rekonstrukce.

Před vlastním formátováním byste opět měli nahlédnout do kapitoly o speciálních nastaveních
mke2fs.

Ve chvíli, kdy vám zařízení RAID funguje, jej můžete zastavovat a spouštět pomocí příkazu:

```
raidstop /dev/md0
```

nebo:

```
raidstart /dev/md0
```

Pomocí programu mdadm jej můžete zastavit příkazem:

mdadm -S /dev/md0

a znovu spustit příkazem:

mdadm -R /dev/md0

Místo vkládání těchto příkazů do init souborů a tisíců restartů, jen abyste zajistili spouštění pole, čtěte dále a raději zprovozněte jejich autodetekci.

Perzistentní superbloky

Za „starých dobrých časů“ používaly raidtools soubor /etc/raidtab k inicializaci pole. To ale znamenalo, že souborový systém, na kterém byl soubor /etc/raidtab, musel být připojen. To je poněkud nešťastné ve chvíli, kdy chcete z RAID zařízení nastartovat systém.

Stejně tak tento starý přístup vedl ke komplikacím v situacích, kdy jste připojovali souborové systé-my nad zařízeními typu RAID. Nebylo možné je umístit jako obvykle do souboru /etc/fstab, ale musely být připojovány přes init skripty.

Perzistentní superbloky tento problém řeší. Jakmile je pole vytvořeno s volbou persistent-super-block v souboru /etc/raidtab, je zapsán na začátek všech disků v poli speciální superblok. Umožňuje tak jádru načíst konfigurace RAID zařízení přímo ze samotných disků, místo načítání nějakého konfiguračního souboru, který nemusí být stále k dispozici.

I tak byste ale měli udržovat konzistentní soubor /etc/raidtab, kvůli budoucí možnosti pole znovu vytvořit.Perzistentní superblok je nutný, pokud chcete používat autodetekci RAID zařízení v době starto-vání systému. Další informace najdete v kapitole Autodetekce.

Velikosti bloků (chunk sizes)

Velikosti bloků (chunk sizes) si jistě zaslouží bližší vysvětlení. Nikdy není možné zapisovat úplně paralelně na skupinu disků. Pokud máte dva disky a chcete zapsat jeden bajt, museli byste zapsat 4 bity na každý z nich. Ve skutečnosti každý druhý bit by skončil na disku 0 a zbylé na disku 1. Současný hardware něco takového prostě nepodporuje. Místo toho se volí nějaká velikost bloku, kterou je definována nejmenší „atomická“ množina dat, která může být na disk zapsána. Zápis 16 kB dat při velikosti bloku 4 kB zajistí, že první a třetí čtyřkilobajtový blok bude zapsán na první disk a druhý a čtvrtý blok na druhý disk (v případě RAID-0 se dvěma disky). Je patrné, že při zapi
sování velkých souborů vzniká menší provozní zátěž s nastavením velkých bloků, zatímco u pole

s většinou malých souborů získáte větší výkon při nastavení menších bloků.Velikost bloku musí být nastavena pro všechny úrovně RAID, včetně lineárního. Nicméně u line-árního módu se na velikost bloku nebere ohled.
Pro optimální výkon byste měli provést pár pokusů s různými hodnotami, stejně jako s velikostí

bloku pro souborový systém, který na pole umístíte.Parametr chunk-size v souboru /etc/raidtab se udává v kilobajtech, takže hodnota „4“ zname-ná „4 kB“.

RAID-0

Data jsou zapisována na disky v poli „vesměs“ paralelně. Ve skutečnosti probíhá zápis po defino

vaných blocích (chunk-size*)* na každý disk sériově.Jestliže nastavíte velikost bloku na 4 kB a budete zapisovat 16 kB na pole složené ze tří disků,pak RAID systém zapíše 4 kB na disky 0, 1 a 2 paralelně a poté zbylé 4 kB dopíše na disk 0.

Optimální počáteční hodnota pro většinu polí je 32 kB. Nicméně ta správná hodnota závisí na počtu zapojených disků, obsahu souborového systému pole a mnoha dalších faktorech. Pro ide
ální výkon je třeba trochu experimentovat.

RAID-0 s ext2

Následujícím tipem přispěl Michael (*michael@freenet-ag.de*):U souborového systému ext2 se více diskové aktivity vyskytuje na začátku skupiny bloků. U jed-
noho disku to ničemu nevadí, ale RAID-0 se může výrazně zpomalit, pokud všechny skupiny bloků začínají na stejném disku. Například:
Pokud máte bloky na RAID zařízení nastaveny na 4 kB a bloky souborového systému také na 4

kB, pak každý blok souborového systému bude umístěn v jednom bloku RAID. Se dvěma diskyje velikost stripu (proužku) 2*4 kB = 8 kB. Výchozí velikost skupiny bloků je 32 768 bloků, takževšechny skupiny bloků začínají na disku 0, což se může stát

úzkým místem snižujícím celkovouvýkonnost pole. Naneštěstí se velikost skupiny bloků dá nastavovat pouze po osmi blocích (32 kBpři použití bloků o velikosti 4 kB), takže se nemůžete problému vyhnout ani úpravou velikostiskupiny bloků nastavené parametrem -g programu mkfs.

Přidáním disku bude velikost stripu 12 kB, takže první skupina bloků bude začínat na disku 0, druhá na disku 2 a třetí na disku 1. Zátěž způsobená aktivitou disku při čtení začátku skupiny bloků se rozloží mezi všechny disky.

Pokud nemůžete přidat další disk, zkuste nastavit velikost bloku na 32 kB. Velikost stripu pak bude 64 kB. Protože můžete nastavovat velikost skupiny bloků pouze po osmi blocích (32 kB), nastavení velikosti skupiny bloků na 32 760 váš problém vyřeší.

Navíc se může stát, že hranice skupiny bloků budou končit na hranicích stripů. To sice není pro blém u výše zmíněných příkladů, ale může se to snadno stát, pokud máte větší velikosti stripů.

RAID-1

Velikost bloků nemá žádný dopad na způsob zápisu na pole, protože stejně musí být všechna data zapsána na všechny disky. Nicméně u čtení udává velikost bloku, kolik dat se má sériově číst z patřičných disků. Ovladač RAID má kompletní svobodu ve výběru disku, ze kterého bude číst, protože všechny aktivní disky v poli obsahují stejné informace. Uvedené vlastnosti využívá ovladač RAID ke zrychlení průměrné přístupové doby tím, že si vybírá pro operaci čtení ten nejdostupnější disk.

RAID-4

Ve chvíli, kdy dojde k zápisu na pole typu RAID-4, musí být také aktualizována informace o pari-tě na paritním disku.Velikost bloku ovlivňuje čtení stejně jako u RAID-0, protože čtení probíhá stejným způsobem.

RAID-5

Na čtení u pole typu RAID-5 má velikost bloku stejný vliv jako u úrovně RAID-0. Zápis na RAID--5 je ale trochu komplikovanější: Když je na pole úrovně RAID-5 zapsán jeden blok, musí být odpovídající blok s paritou aktualizován také. Aktualizace paritního bloku vyžaduje:

Buď původní blok, nový blok a starý paritní blok.
Nebo všechny bloky (kromě paritního bloku) daného stripu.

Ovladač RAID si sám vybere nejsnazší způsob pro aktualizaci bloku podle průběhu zápisu. Přiro-zeně, pokud váš server má dostatek paměti a/nebo zápis je snadný a lineární, pak aktualizace paritních bloků způsobí pouze jeden zápis navíc, který musí projít sběrnicí (stejně jako u RAID--1). Vlastní výpočet parity je velmi efektivní, a i když samozřejmě zatěžuje procesor systému, tak je v podstatě zanedbatelný. Pokud jsou ale zápisy malé a rozložené po celém disku, tak ovladač RAID bude muset kvůli výpočtu paritního bloku skoro vždycky přečíst všechny nedotčené bloky z každého stripu, do kterého je zapisováno. To způsobí dodatečné zatížení sběrnice a zpomalí odezvu systému kvůli násobnému čtení.

Rozumná velikost bloku pro RAID-5 je 128 kB, ale stejně jako v ostatních případech se jedná o záležitost, kterou je třeba nejdříve vyzkoušet.

Přečtěte si také kapitolu o speciálních nastaveních programu mke2fs. I tam můžete ovlivnit výkonnost pole s RAID-5.

Parametry pro mke2fs

Pro nastavení RAID-4 nebo RAID-5 můžete použít speciální parametr programu mke2fs. Jde o parametr -R stride=nn, který umožní lépe umístit specifické datové struktury souborového systému ext2 na RAID zařízení. Jestliže je velikost bloku nastavena na 32 kB, pak bude 32 kB sekvenčních dat uloženo na jednom disku. Pokud nastavíte vytvoření souborového systému ext2 s bloky o velikosti 4 kB, bude na jednom bloku pole uloženo osm bloků souborového systému. Informaci o velikosti bloku sou-borového systému můžete předat nástroji mke2fs ve chvíli jeho vytváření:

```
mke2fs -b 4096 -R stride=8 /dev/md0
```

Výkonnost RAID-{4,5} je velmi ovlivněna uvedenou volbou. Nejsem si jistý, jak moc parametr ovlivňuje výkonnost ostatních úrovní RAID. Pokud někdo má informace o dopadech na jejich výkonnost, prosím, pošlete mi je.

Velikost bloku nastavená programem ext2fs výrazně ovlivňuje výkonnost souborového systému. Měli byste vždy nastavit velikost bloku na 4 kB na souborových systémech větších než několik stovek megabajtů. A to i v případě, že na něm používáte velké množství malých souborů.

Detekce, dotazování a testování

Následující kapitola pojednává o soužití se softwarovým RAID, to znamená o komunikaci a hraní s ním.
Uvědomte si ale, že při manipulaci s md zařízeními vždy pracujete s celými souborovými systémy. Takže musíte být velmi opatrní, i když máte v poli nějakou redundanci, která zabezpečuje vaše soubory.

Detekování vadného disku

Nejedná se o žádnou záhadu. Stačí rychlý pohled do standardních logů a statických souborů
a všimnete si, že disk je poškozený.Soubor /var/log/messages je už z povinnosti vždycky zaplněn velkým množstvím chybovýchzpráv. Když ale dojde k havárii disku, zaplní se ještě větším množstvím chybových zpráv jádra.Pro masochisty je zde několik ošklivých příkladů:

kernel: scsi0 channel 0 : resetting for second half of retries. kernel: SCSI bus is being reset for host 0 channel 0. kernel: scsi0: Sending Bus Device Reset CCB #2666 to Target 0 kernel: scsi0: Bus Device Reset CCB #2666 to Target 0 Completed kernel: scsi : aborting command due to timeout : pid 2649, scsi0, channel 0, id 0, lun 0 Write (6) 18 33 11 24 00 kernel: scsi0: Aborting CCB #2669 to Target 0 kernel: SCSI host 0 channel 0 reset (pid 2644) timed out - trying harder kernel: SCSI bus is being reset for host 0 channel 0. kernel: scsi0: CCB #2669 to Target 0 Aborted kernel: scsi0: Resetting BusLogic BT-958 due to Target 0 kernel: scsi0: *** BusLogic BT-958 Initialized Successfully *** Nejčastěji vypadá chyba disku takto: kernel: hde: read_intr: error=0x10 { SectorIdNotFound }, CHS=31563/14/35, sector=0 kernel: hde: read_intr: status=0x59 { DriveReady SeekComplete DataRequest Error }

A jak se asi dá předpokládat, soubor /proc/mdstat prozradí chybu následujícím obsahem:

Personalities : [linear] [raid0] [raid1] [translucent]
read_ahead not set
md7 : active raid1 sdc9[0] sdd5[8] 32000 blocks [2/1] [U_]

Dále v této kapitole se dozvíte, jak monitorovat RAID pomocí nástroje mdadm, takže můžete být včas upozorněni na chybu disku. Teď je ale čas si říci něco o obsahu souboru /proc/mdstat.

Dotazování se na stav pole

Vždycky si můžete prohlédnout obsah souboru /proc/mdstat. Nemůže to uškodit. Vysvětleme si, jak jeho obsah číst. Tak například:

Personalities : [raid1] read_ahead 1024 sectors md5 : active raid1 sdb5[1]
sda5[0]
4200896 blocks [2/2] [UU]

md6 : active raid1 sdb6[1] sda6[0] 2104384 blocks [2/2] [UU]

md7 : active raid1 sdb7[1] sda7[0] 2104384 blocks [2/2] [UU]
md2 : active raid1 sdc7[1] sdd8[2] sde5[0] 1052160 blocks [2/2] [UU]

unused devices: none

Pro nalezení záložních zařízení se nejdříve podívejte na hodnoty v hranatých závorkách – [#/#]. První číslo označuje úplný počet zařízení v poli. Řekněme, že je to „n“. Role nebo funkce zaříze-ní v poli je označena za každým zařízením v hranatých závorkách, [#]. Jakékoli zařízení, jehož role je větší nebo rovna „n“, je záložní zařízení. 0, 1, .., n-1 jsou aktivní zařízení v poli.

V případě výpadku disku bude vadné zařízení označeno jako (F) hned za označením jeho role [#]. Záložní disk, který nahradí chybný disk, bude ten s nejnižším číslem role nebo s větším, než má disk označený (F). Jakmile doběhne synchronizace, tak se čísla rolí zařízení opět prohodí zpět.

Pořadí, v jakém se zařízení objevují v souboru /proc/mdstat, nemá žádný význam. Nakonec si zapamatujte, že vždycky můžete ověřit stav pole pomocí raidtools nebo mdadm.

mdadm --detail /dev/mdx lsraid -a /dev/mdx

Příkazy vám ukážou jasně a přehledně, která zařízení jsou záložní a která vadná.

Simulace havárie disku

Plánujete-li používat RAID kvůli toleranci výpadku jednoho disku, měli byste také otestovat vaši

konfiguraci a ověřit, že opravdu funguje. Jak ale chybu disku simulovat?Jednoduše řečeno, nejde to. Možná pokud byste byli

ochotni zaseknout požární sekeru do disku,u kterého chcete „simulovat“ chybu. Nikdy nemůžete vědět, co se stane, pokud vám selže disk.Díky elektřině se může poškodit i sběrnice, ke které je připojený, což způsobí nedostupnosti ostatních disků na téže sběrnici. Nikdy jsem o tom sice neslyšel, ale určitě je to možné. Nebomůže disk SCSI/IDE vrstvě jen oznámit chybu čtení/zápisu, což zajistí Raidu pohodlné vypořádá-ní se s nastalou situací. Naštěstí je častější druhá varianta.

Připomeňme si, že musíte mít RAID-{1,4,5}, aby bylo možné tolerovat výpadek disku. Lineární mód nebo RAID-0 bude úplně nedostupný, pokud jeden z disků nebude funkční.

Hardwarová simulace

Pro simulaci chyby disku stačí odpojit kabel disku. To byste měli udělat při vypnutém systému. Nemá smysl hazardovat, pokud chcete jen vyzkoušet, že vaše data budou dostupná i s méně disky, než je normální počet. Vypněte systém, odpojte disk a zase systém spusťte.

Pro ověření stavu se podívejte do souboru /proc/mdstat a zkontrolujete systémový log. Fungu-je to?Chybné disky by se měly objevit v souboru /proc/mdstat se značkou (F). Uživatelé programumdadm by měli vidět stav zařízení jako *faulty* (chybný, poškozený).

Jakmile zařízení znovu připojíte (nezapomeňte, že při vypnutém systému), můžete zařízení znovupřidat k RAID příkazem raidhotadd.

Softwarová simulace

Novější verze raidtool obsahují příkaz raidsetfaulty. S tímto příkazem můžete jednoduše simulo-vat chybu disku, aniž byste jej museli odpojit.Spuštění příkazu:

```
raidsetfaulty /dev/md1 /dev/sdc2
```

by mělo být dostatečné pro nastavení disku /dev/sdc2 v poli /dev/md1 jako chybného. Při pou-žití mdadm stačí spustit:

```
mdadm --manage --set-faulty /dev/md1 /dev/sdc2
```

Tím by se měly věci dát do pohybu a začne zábava. Nejprve byste měli vidět v systémovém logu něco podobného prvnímu řádku. Pokud máte nastavený záložní disk, měl by se objevit i záznam podobný druhému řádku:

```
kernel: raid1: Disk failure on sdc2, disabling device. kernel: md1: resyncing spare disk sdb7 to replace failed disk
```

Kontrola souboru /proc/mdstat ukáže pole v degradovaném režimu. Při použití záložního diskubyste měli vidět průběh rekonstrukce.Další nový nástroj z nejnovějších raidtools je lsraid. Vyzkoušejte:

```
lsraid -a /dev/md1
```

a jste-li uživatelé *mdadm,* můžete vyzkoušet:

```
mdadm --detail /dev/md1
```

a užijte si jejich výstupu.Nyní jste viděli, co se stane při výpadku disku. Pojďme dát věci zase do pořádku.Nejdříve vyjměte poškozený disk z pole, a to příkazem:

```
raidhotremove /dev/md1 /dev/sdc2
```

uživatelé mdadm použijí:

```
mdadm /dev/md1 -r /dev/sdc2
```

Poznamenejme, že raidhotremove nemůže vyjmout aktivní disk z běžícího pole. Ze zřejmých důvodů lze takto odebrat pouze chybný disk (spuštění raidstop a odpojení zařízení nepomůže). Máte tedy zařízení /dev/md1, které právě přišlo o disk. Může být buď ve stavu degradovaného pole nebo možná uprostřed procesu rekonstrukce. Dříve než začnete stav vracet k normálu, počkejte, než zotavení skončí.

Výlet tedy skončí ve chvíli, kdy vrátíte zařízení /dev/sdc2 zpátky do svého domova.

```
raidhotadd /dev/md1 /dev/sdc2
```

Jako vždy můžete použít mdadm namísto raidtools. Mělo by se jednat o příkaz:

```
mdadm /dev/md1 -a /dev/sdc2
```

Můžete si všimnout, že jakmile se ztracený syn vrátí zpět, začne být okamžitě aktivním členem pole /dev/md1. Samozřejmě pouze v případě, že je to nutné. Pokud ne, tak bude označen jako záložní disk. Jde hlavně o co nejjednodušší správu.

Simulace poškození dat

RAID (ať hardwarový nebo softwarový) předpokládá, že zápis dat na disk, který nevrátí chybu, je zápis úspěšný. Takže jestliže váš disk poškodí vaše data, aniž by oznámil chybu, stanou se data nepoužitelná. Jedná se samozřejmě o velmi málo pravděpodobný scénář, ale je to možné a pravděpodobně to skončí poškozeným souborovým systémem.

RAID neumí, ani k tomu není určen, hlídat chybný zápis dat na médium. Takže ani nemá smysl účelově poškodit data na disku (například použitím programu dd) a zkoušet, jak se s tím RAID vypořádá. Je hodně pravděpodobné (pokud nepoškodíte přímo superblok Raidu), že vrstva Raidu
ani nezjistí poškozená data, ale souborový systém na RAID zařízení poškozený bude. Takhle ale mají věci fungovat. RAID neposkytuje žádnou záruku integrity dat, pouze umožňuje vaše data zachovat v případě výpadku disku (samozřejmě s úrovní Raidu jedna nebo vyšší).

Monitorování pole

Můžete spustit mdadm jako démona monitorujícího stav pole. V případě potřeby zašle mdadm e-mail administrátorovi systému, že se na polích vyskytují problémy. Je možné jej také nastavit do módu pro spouštění připravených akcí, což může v případě výpadku dát například disku druhou šanci tím, že jej odpojí a zpět vloží do pole. Nekritické pády disku tak mohou být vyřešeny automaticky.

Podívejme se na jednoduchý příklad. Příkaz:

```
mdadm --monitor --mail=root@localhost --delay=1800 /dev/md2
```

spustí démona mdadm, který bude monitorovat pole /dev/md2. Parametr --delay znamená časo-vý interval kontroly 1 800 vteřin. Kritické události a zásadní chyby by měly být zaslány na e-mail administrátora systému. Monitorování RAID je opravdu jednoduché.

Můžete také použít parametry --program a --alert, které zajistí spuštění definovaného programu v případě nějaké události.

> Poznámka
>
> Démon mdadm nikdy neskončí, dokud budou existovat pole, na která dohlíží. Nezapo-
> meňte, že spouštíte démona, ne obyčejný příkaz.

Používání programu mdadm pro monitorování pole RAID je jednoduché a efektivní. Nicméně jsou zde stále základní otázky vyplývající z takového typu monitorování. Například co se stane, když démon mdadm přestane fungovat? Abyste předešli takovým problémům, měli byste se porozhlédnout po „opravdových“ monitorovacích nástrojích. Existuje mnoho volně dostupných programů, open-source programů a komerčních řešení, která jsou použitelná pro sledování soft-warového RAID na GNU/Linuxu. Vyhledávání na portálu *FreshMeat* (*http://freshmeat.net/*) vám vrátí rozumný počet výsledků.

Nastavování, ladění a hledání chyb

raid-level a raidtab

Některé linuxové distribuce, jako třeba RedHat 8.0 a možná i jiné, mají chybu ve svých init skrip-tech. Nepodaří se jim spustit pole RAID při startu systému, jestliže v souboru /etc/raidtab jsou mezery nebo tabelátory před klíčovým slovem raid-level.

Problém se dá jednoduše obejít tím, že zajistíte, aby se slovo raid-level objevilo na úplném začátku řádku, bez jakýchkoli úvodních mezer.

Autodetekce

Autodetekce umožňuje, aby bylo zařízení RAID jádrem automaticky rozpoznáno v době startová
ní systému hned poté, co bylo rozpoznáno normální diskové zařízení.
To vyžaduje několik věcí:

Zkontrolujte, že máte podporu autodetekce v jádře.
Musíte vytvořit RAID zařízení s použitím perzistentních superbloků.
Typy diskových oddílů RAID zařízení musí být nastaveny na 0xFD (použijte fdisk a nastav-te typ na „fd“).

> Upozornění
>
> Ujistěte se, že RAID zařízení v době změny typu diskového oddílu NEBĚŽÍ. Pro zastavení
> zařízení použijete raidstop /dev/md0.

Jestliže podmínky 1, 2 a 3 splňujete, měla by být autodetekce nastavená. Zkuste restartovat systém. Po startu systému si vypište

soubor /proc/mdstat, který by vám měl oznámit, že zařízení RAID je spuštěné.

Během startování systému můžete vidět zprávy podobné těmto:

Oct 22 00:51:59 malthe kernel: SCSI device sdg: hdwr sector= 512 bytes. Sectors= 12657717 [6180 MB] [6.2 GB] Oct 22 00:51:59 malthe kernel: Partition check: Oct 22 00:51:59 malthe kernel: sda: sda1 sda2 sda3 sda4 Oct 22 00:51:59 malthe kernel: sdb: sdb1 sdb2 Oct 22 00:51:59 malthe kernel: sdc: sdc1 sdc2 Oct 22 00:51:59 malthe kernel: sdd: sdd1 sdd2 Oct 22 00:51:59 malthe kernel: sde: sde1 sde2 Oct 22 00:51:59 malthe kernel: sdf: sdf1 sdf2 Oct 22 00:51:59 malthe kernel: sdg: sdg1 sdg2 Oct 22 00:51:59 malthe kernel: autodetecting RAID arrays Oct 22 00:51:59 malthe kernel: (read) sdb1's sb offset: 6199872 Oct 22 00:51:59 malthe kernel: bind<sdb1,1> Oct 22 00:51:59 malthe kernel: (read) sdc1's sb offset: 6199872 Oct 22 00:51:59 malthe kernel: bind<sdc1,2> Oct 22 00:51:59 malthe kernel: (read) sdd1's sb offset: 6199872 Oct 22 00:51:59 malthe kernel: bind<sdd1,3> Oct 22 00:51:59 malthe kernel: (read) sde1's sb offset: 6199872 Oct 22 00:51:59 malthe kernel: bind<sde1,4> Oct 22 00:51:59 malthe kernel: (read) sdf1's sb offset: 6205376 Oct 22 00:51:59 malthe kernel: bind<sdf1,5> Oct 22 00:51:59 malthe kernel: (read) sdg1's sb offset: 6205376 Oct 22 00:51:59 malthe kernel: bind<sdg1,6> Oct 22 00:51:59 malthe kernel: autorunning md0 Oct 22 00:51:59 malthe kernel: running: <sdg1><sdf1><sde1><sdd1><sdc1><sdb1> Oct 22 00:51:59 malthe kernel: now! Oct 22 00:51:59 malthe kernel: md: md0: raid array is not clean -starting background reconstruction

Jde o výstup z autodetekce pole typu RAID-5, které nebylo správně zastaveno (resp. počítač zamrzl). Rekonstrukce byla automaticky zahájena. Připojení zařízení je naprosto bezpečné, proto-že rekonstrukce je transparentní a všechna data jsou konzistentní (pouze paritní informace není konzistentní, ale ta není zapotřebí, dokud zařízení neselže).

Automaticky spouštěná zařízení jsou také automaticky vypínána při zastavování systému. Nedě-lejte si starosti s init skripty. Prostě používejte zařízení /dev/md stejně jako ostatní zařízení /dev/sd nebo /dev/hd.

Ano, opravdu je to tak snadné.
Můžete se také podívat do init skriptů na příkazy raidstart/raidstop. Často je můžete nalézt ve skriptech starších RedHat distribucí. Používají se pro starší RAID a v novějších verzích RAID s autodetekcí nemají význam. Klidně můžete příslušné řádky smazat a všechno bude v pořádku.

Startování systému ze zařízení typu RAID

Je několik možností, jak nastavit systém, aby připojoval hlavní souborový systém z RAID zaříze-ní. Některé distribuce umožňují nastavit RAID již při instalaci systému, což je zároveň i nejjedno-dušší způsob, jak RAID jednoduše zprovoznit.

Novější LILO umí pracovat se zařízeními RAID-1, a tak může být jádro načteno z RAID zařízení již v době startu systému. LILO správně zapíše i boot záznamy na všechny disky v poli, takže může-te spustit systém i při výpadku primárního disku.

Používáte-li GRUB místo zavaděče LILO, pak jej stačí spustit a nastavit druhý (ale i třetí nebo čtvr-tý, ..) disk v poli RAID-1, ze kterého chcete startovat systém jako hlavní (root) zařízení, a spustit setup. To je vše.

Například na poli složeném z disků /dev/hda1 a /dev/hdc1, kde by z obou oddílů mělo být možné spustit systém, byste měli udělat následující:

```
grub
grub>device (hd0) /dev/hdc grub>root (hd0,0) grub>setup (hd0)
```

Někteří uživatelé mají s výše uvedeným problémy, projevující se nemožností spustit systém s oběma disky, ale pouze s jedním připojeným diskem. Každopádně uvedený postup spuštěný s připojenými oběma disky umožní systému startovat jen s jedním diskem nebo z pole RAID-1.

Jiný způsob, jak zajistit, že váš systém vždycky nastartuje, je vytvořit startovací disketu ve chvíli, kdy máte všechno nastavení hotové. Vždy můžete systém spustit z diskety, i když je disk, na kte-rém je umístěn souborový systém /boot, poškozený. Na systémech RedHat (a z něj odvozených distribucí) můžete disketu vytvořit příkazem mkbootdisk.

Hlavní souborový systém na RAID

Abyste měli systém startující z pole RAID, musíte mít připojený hlavní souborový systém (root, /) na RAID zařízení. Dále jsou popsány dva způsoby, jak toho dosáhnout. První metoda předpoklá-dá, že instalujete systém na normální oddíl, a pak – když je instalace hotová – přesunete obsah z hlavního ne-RAID souborového systému na RAID zařízení. Připomeňme, že u novějších distri-bucí tento způsob již není nutný, protože distribuce umí instalovat systém přímo na RAID zaříze-ní (a tato zařízení během instalace vytvořit). Nicméně postup se může hodit, pokud migrujete sou-časný systém na RAID.

Způsob 1

Předpokladem je, že máte jeden volný disk (který není součástí nastavovaného pole RAID), na který můžete systém nainstalovat.

Nejprve nainstalujte normální systém na volný disk.
Připravte si jádro, které plánujete používat, jeho úpravy pro RAID a potřebné nástroje. Ujis-těte se, že váš systém nastartuje s tímto novým jádrem. Přesvědčte se, že podpora RAID je v jádře a že není nahrána jako modul.
Nyní byste měli nastavit a vytvořit zařízení RAID, na které chcete umístit hlavní souborový systém. Jedná se o standardní postup popsaný na jiných místech návodu.
Zkontrolujte, že je všechno v pořádku, a zkuste spustit systém. RAID zařízení by se mělo spustit již při startu.
Vytvořte souborový systém na novém poli (použitím mke2fs) a připojte jej do adresáře

/mnt/newroot.

- Nyní zkopírujte obsah současného hlavního souborového systému (z volného disku) do nového hlavního souborového systému (na diskové pole). Existuje mnoho způsobů, jak toho docílit. Jedním z nich je:

  cd /find . -xdev | cpio -pm /mnt/newroot

  Další způsob může zkopírovat vše z adresáře / do adresáře /mnt/newroot:

  cp -ax / /mnt/newroot

Měli byste upravit soubor /mnt/newroot/fstab a nastavit správné zařízení (hlavní zaříze-ní /dev/md?) pro hlavní souborový systém.
Nyní odpojte současný souborový systém /boot a připojte místo něj /mnt/newroot/boot. Vyžaduje to zavaděč LILO, aby mohl korektně proběhnout následující krok.
Aktualizujte soubor /mnt/newroot/etc/lilo.conf, aby ukazoval na správná zařízení. Oddíl /boot musí být stále normální disk (ne RAID zařízení), ale hlavní souborový systém by měl odkazovat na vaše nové RAID zařízení. Jakmile máte hotovo, spusťte:

lilo -r /mnt/newroot

Příkaz by měl skončit bez chybových hlášení.

- Restartujte systém a sledujte, jestli vše funguje, jak očekáváte.

Pokud uváděný postup zkoušíte s IDE disky, ujistěte se, že všechny disky jsou v BIOSu nastave-ny na „auto-detect“, aby byl BIOS, v případě chybějícího disku, schopen počítač spustit.

Způsob 2

Varianta vyžaduje, aby vaše jádro a raidtools rozumělo parametru failed-disk v souboru /etc/raidtab. Jestliže používáte opravdu staré jádro, tak budete muset pravděpodobně nejdříve aktualizovat jádro a potřebné nástroje.

Způsob můžete použít, *pouze* pokud používáte RAID úrovně 1 a vyšší, protože je využíváno pole v „degradovaném režimu“, což je možné pouze pro RAID s redundancí. Hlavní myšlenka je insta-lovat systém na disk, který záměrně označíte v poli jako vadný. Pak překopírujete systém na zaří-zení běžící v degradovaném režimu a nakonec necháte RAID použít dále nepotřebný „instalační disk“. Spustíte pole v nedegradovaném režimu a smažete starou instalaci.

Nejdříve nainstalujte normální systém na jeden disk (později se stane součástí pole). Je důležité, aby tento disk (nebo oddíl) nebyl ten nejmenší. Jinak ho nebude možné později přiřadit do diskového pole.
Pak si připravte jádro, jeho úpravy, potřebné nástroje atd. Víte, jak to chodí. Nechte systém naběhnout s novým jádrem s potřebnou podporou Raidu, zakompilovanou přímo do jádra.
Nastavte RAID se současným hlavním oddílem jako failed-disk v souboru /etc/raidtab. Nezadávejte jej jako první disk v souboru /etc/raidtab. Způsobilo by to potíže při spouš-

tění RAID. Vytvořte zařízení RAID a umístěte na něj souborový systém. Při použití mdadm můžete vytvořit degradované pole pouhým spuštěním podobného příkazu:

mdadm -C /dev/md0 --level raid1 --raid-disks 2 --missing /dev/hdc1

Všimněte si parametru **--**missing (*chybějící*).

Zkuste restartovat systém a přesvědčte se, že se RAID spustil, jak by měl.
Překopírujte systémové soubory a přenastavte systém, aby používal RAID jako hlavní zaří-zení (stejně jako v předchozím způsobu).
Jakmile se váš systém korektně spustí z RAID zařízení, můžete upravit soubor /etc/raid-tab a změnit předchozí failed-disk na normální raid-disk. Teď použijte raidhotadd pro při-dání disku k poli.
V tuto chvíli byste měli mít systém, který bude schopen se spustit z nedegradovaného pole RAID.

Spouštění systému z Raidu

Aby jádro bylo schopné připojit hlavní souborový systém, musí být veškerá podpora pro zaříze-ní, na kterém je tento systém, v jádře přítomna. Takže abyste mohli připojit hlavní souborový systém na RAID zařízení, *musíte* mít v jádře podporu pro RAID.

Normální způsob, jak zajistit, že jádro umí pracovat s RAID zařízením, je jednoduše zkompilovat veškerou potřebnou podporu pro RAID přímo do jádra. Ujistěte se, že je podpora zkompilována přímo *do* jádra a *ne* pouze jako moduly. Jádro nemůže načítat moduly (z hlavního souborového systému), dokud není souborový systém připojený.

Nicméně RedHat od verze 6.0 obsahuje jádro, které má podporu nového Raidu jako modulu. Dále si popíšeme, jak můžete mít s pomocí *initrd* standardní RedHat jádro a stále moci startovat systém ze zařízení typu RAID. Initrd je v dnešních distribucích obvykle přítomen.

Startování systému z Raidu jako modulu

Abyste dosáhli požadovaného výsledku, budete muset nastavit LILO tak, aby používalo RAM-disk. Použijte příkaz mkinitrd pro vytvoření RAM-disku, který bude obsahovat všechny moduly potřeb-né k připojení hlavního oddílu. Poslouží k tomu následující příkaz:

mkinitrd --with=<module> <ramdisk name> <kernel>

Například:

mkinitrd --preload raid5 --with=raid5 raid-ramdisk 2.2.5-22

To zajistí, že potřebný RAID modul bude jádru k dispozici pro připojení hlavního oddílu již ve chvíli startu systému.

Modulární RAID na Debian GNU/Linuxu po přesunutí na RAID

Uživatelé Debianu mohou narazit na problémy při používání initrd pro připojení jejich hlavního oddílu ze zařízení RAID, pokud migrovali standardní ne RAID instalaci Debianu na hlavní oddíl na RAID zařízení.

Jestliže váš systém odmítá připojit hlavní souborový systém při startu systému (uvidíte hlášení „kernel panic“), může být problém v tom, že souborový systém initrd nemá potřebnou podporu pro připojení hlavního souborového systému z RAID zařízení.
Debian vytváří svůj initrd.img na základě předpokladu, že hlavní souborový systém, který se bude připojovat, je ten právě aktuální. To obvykle skončí jako „kernel panic“ ve chvíli, kdy je hlav-ní souborový systém přesunut na RAID zařízení a vy se pokusíte spustit systém s nezměněným obrazem initrd.img. Řešením je použít příkaz mkinitrd, ale určit cílový hlavní souborový systém. Například následující sada příkazů by měla na systému Debian vytvořit a nastavit nový initrd:

% mkinitrd -r /dev/md0 -o /boot/initrd.img-2.4.22raid % mv /initrd.img /initrd.img-nonraid % ln -s /boot/initrd.img-raid /initrd.img”

Přesun RedHat systému bez Raidu na softwarový RAID

Kapitolu napsal a do návodu poskytl Mark Price, IBM. Od své první verze prošel text drobnými změnami. Obsahuje technické detaily, jak převést linuxový systém bez Raidu pro použití se soft-warovým Raidem. Postup byl vyzkoušen na systému RedHat 7.1, ale měl by být aplikovatelný na jakýkoli systém, který podporuje softwarový RAID (zařízení md).

Poznámka

Následující informace jsou poskytovány „tak jak jsou“ bez nároku na jakoukoli záruku, ať už vyslovenou či předpokládanou. Návod můžete použít dobrovolně na vlastní riziko a nikdo jiný nebude zodpovědný za jakoukoli škodu způsobenou jeho použitím.

Testovací systém před převodem

Testovací systém obsahuje dva SCSI disky, sda a sdb, stejné velikosti. Pro potřeby testu jsem nasta-vil oba disky tak, aby měly stejné rozdělení oddílů. Použil jsem fdisk, abych měl jistotu, že počet bloků na každém oddíle bude stejný.

DEVICE	MOUNTPOINT	SIZE	DEVICE	MOUNTPOINT	SIZE
/dev/sda1	/	2048MB	/dev/sdb1		2048MB
/dev/sda2	/boot	80MB	/dev/sdb2		80MB
/dev/sda3	/var/	100MB	/dev/sdb3		100MB
/dev/sda4	SWAP	1024MB	/dev/sdb4	SWAP	1024MB

Na tomto snadném příkladu nastavíme jednoduché zrcadlení RAID-1, které vyžaduje pouze dva fyzické disky.

Krok-1 – spuste systém ze záchranného CD/diskety

Instalační CD distribuce RedHat obsahuje záchranný režim, který spustí Linux z CD a připojí jaký koli souborový systém, který na vašich discích najde.
Na příkazové řádce při startu napište:

lilo: linux rescue

U výše popsané konfigurace by se měl instalátor zeptat, na kterém disku je hlavní souborovýsystém. Vyberte sda.Instalátor připojí váš souborový systém následujícím způsobem:

DEVICE MOUNTPOINT TEMPORARY MOUNT
POINT /dev/sda1 / /mnt/sysimage /dev/sda2 /boot /mnt/sysimage/boot /dev/sda3 /var /mnt/sysimage/var /dev/sda6 /home /mnt/sysimage/home

Poznámka

Mějte, prosím, na paměti, že ostatní distribuce mohou připojit váš souborový systém do jiného adresáře nebo mohou vyžadovat, abyste jej připojili ručně.

Krok-2 – vytvořte soubor /etc/raidtab

Vytvořte soubor /mnt/sysimage/etc/raidtab (nebo kdekoli je ve skutečnosti adresář /etc při-pojen). U našeho testovacího systému bude soubor raidtab vypadat takto:

```
raiddev /dev/md0
raid-level                1
nr-raid-disks             2
nr-spare-disks            0
chunk-size                4
persistent-superblock     1
device                    /dev/sda1
raid-disk                 0
device                    /dev/sdb1
raid-disk                 1

raiddev /dev/md1
raid-level                1
nr-raid-disks             2
nr-spare-disks            0
chunk-size                4
persistent-superblock     1
device                    /dev/sda2
raid-disk                 0
device                    /dev/sdb2
raid-disk                 1

raiddev /dev/md2
raid-level                1
nr-raid-disks             2
nr-spare-disks            0
chunk-size                4
persistent-superblock     1
device                    /dev/sda3
raid-disk                 0
device                    /dev/sdb3
raid-disk                 1
```

Poznámka

Je důležité, aby zařízení byla ve správném pořadí. Například že /dev/sda1 je raid-disk 0 a ne raid-disk 1. Říká to ovladači md, že má data synchronizovat z /dev/sda1. Kdyby byl zápis obráceně, začala by synchronizace z disku /dev/sdb1, což by vám zničilo souboro-vý systém.

Nyní zkopírujte soubor raidtab z vašeho skutečného hlavního souborového systému na aktuál-ní hlavní souborový systém:

```
(rescue)# cp /mnt/sysimage/etc/raidtab /etc/raidtab
```

Krok-3 – vytvořte zařízení md

Existují dva způsoby, jak je vytvořit. Buď můžete zkopírovat soubory z adresáře /mnt/sysimage/dev nebo použít mknod pro jejich vytvoření. Zařízení md je blokové zařízení s hlavním číslem 9.

(rescue)# mknod /dev/md0 b 9 0 (rescue)# mknod /dev/md1 b 9 1
(rescue)# mknod /dev/md2 b 9 2

Krok-4 – odpojte souborové systémy

Abyste mohli spustit RAID zařízení a synchronizovat disky, musíte nejdříve odpojit všechny dočasné souborové systémy:

(rescue)# umount /mnt/sysimage/var (rescue)# umount /mnt/sysimage/boot
(rescue)# umount /mnt/sysimage/proc (rescue)# umount /mnt/sysimage

Může se stát, že se vám nepodaří odpojit adresář /mnt/sysimage. Problém může být způsobený záchranným režimem. Měl by se vyřešit tím, že zvolíte připojování oddílů ručně místo automaticky.

Krok-5 – spuste RAID zařízení

Protože na zařízeních /dev/sda1, /dev/sda2 a /dev/sda3 jsou souborové systémy, je nutné si vynutit spuštění RAID zařízení:

(rescue)# mkraid --really-force /dev/md2

Můžete sledovat chování synchronizace průběžným výpisem souboru /proc/mdstat. Ukáže vámstav RAID zařízení a kolik procent zbývá do konce synchronizace.Pokračujte s oddíly /boot a /:

(rescue)# mkraid --really-force /dev/md1 (rescue)# mkraid --really-force /dev/md0

Ovladač md synchronizuje v danou chvíli vždy pouze jeden disk.

Krok-6 – znovu připojte souborové systémy

Připojte nově synchronizované souborové systémy zpět do adresáře /mnt/sysimage:

(rescue)# mount /dev/md0 /mnt/sysimage (rescue)# mount /dev/md1 /mnt/sysimage/boot
(rescue)# mount /dev/md2 /mnt/sysimage/var

Krok-7 – změňte kořenový adresář

Nyní musíte změnit aktuální kořenový adresář na hlavní souborový systém:

(rescue)# chroot /mnt/sysimage

Krok-8 – upravte konfigurační soubory

Musíte adekvátně změnit nastavení zavaděče LILO a souboru /etc/fstab, abyste mohli spouštět systém z md zařízení.

Poznámka

Oddíl /boot nesmí být na RAID zařízení. Hlavní (root*)* zařízení je váš nový md0 disk.
boot=/dev/sda map=/boot/map install=/boot/boot.b prompt timeout=50 message=/boot/message linear default=linux

image=/boot/vmlinuz label=linux read-only root=/dev/md0
Upravte soubor /etc/fstab

/dev/md0	*/*	*ext3*	*defaults*	*1 1*
/dev/md1	/boot	ext3	defaults	1 2
/dev/md2	/var	ext3	defaults	1 2
/dev/sda4	swap	swap	defaults	0 0

Krok-9 – spuste LILO

S upraveným souborem /etc/lilo.conf zohledňujícím nové hlavní zařízení root=/dev/md0 ase zařízením /dev/md1 připojeným jako /boot můžete nyní spustit na změněném kořenovém adre-sáři příkaz /sbin/lilo -v.

Krok-10 – změňte typ diskových oddílů

Typ diskových oddílů na všech oddílech u VŠECH disků, které používá ovladač md, musí být změ-něn na 0xFD.
Pro změnu typu oddílu použijte program fdisk s použitím volby t.Po změně všech požadovaných oddílů použijte volbu w, aby se změny uložily do tabulky disko-vých oddílů.

Krok-11 – změňte velikost souborového systému

Po vytvoření zařízení typu RAID je fyzická velikost oddílu o něco menší, protože druhý superblok je uložený na konci oddílu. V tuto chvíli by restartování systému zkolabovalo s chybou indikující, že je superblok poškozený.

Změňte velikost ještě před restartem. Ujistěte se, že jsou všechna md zařízení, kromě kořenového, odpojena a znovu je připojte pouze pro čtení:

```
(rescue)# mount / -o remount,ro
```

Budete nuceni zkontrolovat všechna md zařízení nástrojem fsck. To je také důvod, proč bylo nutné oddíly znovu připojit pouze pro čtení. Parametr -f donutí nástroj fsck zkontrolovat i čistý soubo-rový systém.

```
(rescue)# e2fsck -f /dev/md0
```

Tento příkaz vygeneruje stejnou chybu ohledně nekonzistentních velikostí a možností poškoze-ného superbloku. Na otázku „Abort?“ odpovězte N.

```
(rescue)# resize2fs /dev/md0
```

Postup zopakujte pro všechna zařízení /dev/md.

Krok-12 – kontrola

Další krok je restart systému. Dříve než ale budete systém restartovat, zkontrolujte následující seznam a ujistěte se, že máte splněné všechny kroky:

Všechna zařízení dokončila synchronizaci. Podívejte se do /proc/mdstat.
Soubor /etc/fstab je upraven tak, že zohledňuje změnu názvů zařízení.
Soubor /etc/lilo.conf je upraven tak, že zohledňuje změnu názvů zařízení.
Přikaz /sbin/lilo byl spuštěn a aktualizoval zavaděč systému.
Jádro má v sobě zkompilovanou podporu jak pro SCSI, tak pro RAID (MD) zařízení
Typy všech diskových oddílů, které jsou součástí zařízení md, byly změněny na 0xfd.
U všech souborových systémů proběhla kontrola (pomocí fsck) a změna velikosti (pomo-cí resize2fs).

Krok-13 – restart

Nyní můžete bezpečně restartovat systém. Jakmile se systém spustí, provede automatické hledánízařízení md (na základě typu diskového oddílu).Váš hlavní souborový systém (root adresář /) by teď měl být zrcadlen.

Sdílení záložních disků mezi více poli

Když používáte nástroj mdadm v monitorovacím režimu, můžete pro různá pole využít jednoho sdíleného záložního disku. Jistě vám to ušetří kapacitu disků bez ztráty komfortu záložních disků. Ve světe softwarového Raidu je to zcela nová, nikdy-předtím-neviděná vlastnost. Pro zajištění oblastí volných disků stačí poskytnout jen jeden volný fyzický disk celé skupině polí.

Nástroj mdadm běžící jako démon kontroluje stav polí v pravidelných intervalech. Jakmile je poškozen disk v poli, které nemá definovaný záložní disk, démon mdadm odebere dostupný záložní disk z jiného pole a vloží jej do pole s chybným diskem. Rekonstrukce degradovaného pole začne jako obvykle okamžitě.

Abyste definovali sdílený záložní disk, použijte při spouštění démona mdadm parametr spare-group.

Nástrahy

Nikdy neměňte oddíly, které jsou součástí běžícího pole! Pokud už musíte změnit tabulku disko-vých oddílů u disku zapojeného v poli, tak nejdříve pole zastavte a pak teprve měňte nastavení oddílů.

Je snadné zapojit příliš mnoho disků na sběrnici. Normální Fast-Wide SCSI sběrnice dokáže zvlád-nout rychlost 10 MB/s, což je méně, než dokáže dnešní disk sám. Vložení šesti takových disků na jednu sběrnici samozřejmě nepřinese očekávaný výkonnostní nárůst. Stejně jednoduché je zahltit i PCI sběrnici. Uvědomte si, že normální 32bitová PCI sběrnice o rychlosti 33 MHz má teoretické maximum propustnosti okolo 133 MB/s. Když uvážíte nějaké nároky na ovládací příkazy a podob-ně, dostanete se ještě o něco níže. Dnešní disky dosahují rychlosti přesahující 30 MB/s, takže jen čtyři takové disky mohou snadno sběrnici zahltit. Při plánování vysokorychlostního RAID systému vždy uvažujte o celé cestě vstupně/výstupních operací. Existují desky s více PCI sběrnicemi, s 64bitovými sběrnicemi o rychlosti 66 MHz a desky se sběrnicemi PCI-X.
Více SCSI řadičů zvýší výkon pouze v případě, že stávající řadiče SCSI jsou takřka plně vytěžová-ny připojenými disky.
Nepoznáte žádný výkonnostní rozdíl při použití dvou řadičů 2940 se dvěma starými SCSI disky oproti případu, kdy budou oba disky na jednom řadiči.

V případě, že zapomenete nastavit perzistentní superblok, nemusí vaše pole dobrovolně nastarto-vat poté, co jej zastavíte. Prostě pole vytvořte znovu a nastavte parametr správně v souboru *raid-tab*. Pamatujte ale, že operace zničí všechna data na poli již uložená.

Špatné pořadí zařízení v souboru raidtab může zapříčinit, že se poli typu RAID-5 nepodaří pro-vést rekonstrukci poté, co byl

odstraněn a znovu vložen disk. Zkuste přesunout první pár zázna-mů „device...“ a „raid-disk“ na konec souboru raidtab.

Rekonstrukce

V případě, že jste pozorně četli předcházející text, měli byste mít velmi dobrou představu o tom, co rekonstrukce degradovaného pole zahrnuje. Shrňme si to:

Vypnout systém.
Vyměnit poškozený disk.
Znovu zapnout systém.
Použít raidhotadd /dev/mdX /dev/sdX na znovu vložení disku do pole.
Dát si kávu, zatímco budete sledovat průběh automatické rekonstrukce.

A to je vše. Většinou je to tedy vše. Pokud nemáte smůlu a vaše pole se stalo nepoužitelné, protože zkola-boval více než jeden redundantní disk. To se může například stát ve chvíli, kdy je několik disků zapojených na jedné sběrnici a pád jednoho disku způsobí pád celé sběrnice. Další disky, ačkoli jinak v pořádku, jsou nedostupné pro RAID vrstvu, protože sběrnice není funkční. Jsou tedy ozna-
čené jako vadné. V poli RAID-5, kde můžete postrádat pouze jeden disk, je ztráta dvou a více disků osudová. Další část je vysvětlením, které mi poskytl Martin Bene a popisuje možnosti zotavení se z výše
popsaného strašidelného scénáře. Zahrnuje použití parametru failed-disk v souboru /etc/raid-tab (takže uživatelé upraveného jádra verze 2.2 by ale měli mít verzi 2.2.10 nebo vyšší).

Zotavení se z pádu více disků

Scénář je následující:

Řadič byl poškozen a tím znepřístupnil dva disky ve stejnou chvíli.
Všechny disky na SCSI sběrnici jsou nedostupné ve chvíli výpadku jednoho disku.
Uvolnily se kabely...

Ve zkratce: Často se stane, že dojde ke *krátkodobému* výpadku několika disků najednou. V tom případě jsou superbloky zařízení RAID nekonzistentní a není možné pole dál používat. Používáte-li nástroj mdadm, můžete nejdříve zkusit:

```
mdadm --assemble --force
```

Pokud ne, tak stále zbývá ještě jedna možnost. Vynutit přepsání superbloků Raidu příkazem mkra-id --force.
Abyste příkaz mohli spustit, musíte mít aktualizovaný soubor /etc/raidtab. Pokud neodpovídá PŘESNĚ zařízením a pořadí původních disků, pravděpodobně nebude příkaz fungovat dle oče-kávání. Za to velmi pravděpodobně přemaže jakákoli data, která jste na discích měli.

Po pokusu spustit pole se podívejte do systémového logu. Uvidíte v něm pro každý superblok počet událostí (*event count*). Obvykle je nejlepší vyjmout disk s nejnižším počtem událostí, to zna-mená ten nejstarší.

Pokud použijete příkaz mkraid bez parametru failed-disk, obnova začne okamžitě a začne pře-
pisovat paritní bloky, což nemusí být přesně to, co v danou chvíli chcete. S parametrem failed-disk můžete přesně určit, který disk chcete označit jako aktivní, a možná vyzkoušet více možností vedoucích k nejlepšímu výsledku. Mimochodem, během zkoušení při-pojujte oddíl jen v režimu pro čtení. Postup byl úspěšně použit alespoň dvěma kolegy, se který-mi jsem v kontaktu.

Výkonnost

Kapitola shrnuje několik výsledků testování rychlosti z opravdových systémů používajících soft-warový RAID. Obsahuje také několik obecných informací o programech testujících výkonnost pole.

Výkonnostní testy byly prováděny programem *bonnie* a ve všech případech se souborem o dva-
krát větší velikosti, než byla fyzická paměť počítače (RAM). Testy měří *pouze* vstupní a výstupní propustnost s jedním velkým souborem. Je to zajímavé pro ty, pro něž je důležitá maximální propustnost při mnoha zápisech a čtení. Nicméně i takové testy nám řeknou jen velmi málo o výkonnosti pole, kdyby bylo použito na news server, webový ser-ver a podobně. Vždy mějte na paměti, že čísla z testů jsou výsledky „syntetického“ programu. Jen několik opravdových programů dělá to, co dělá bonnie, a ačkoli takové výsledky jsou hezké na pohled, tak nikdy nebudou přesně odpovídat realitě cílového systému. A to ani těsně.

Momentálně mám výsledky pouze z mého vlastního počítače. Konfigurace je následující:

Dual Pentium Pro 150 MHz
256 MB RAM (60 MHz EDO)
Třikrát IBM UltraStar 9ES 4.5 GB, SCSI U2W

Adaptec 2940U2W
Jeden disk IBM UltraStar 9ES 4.5 GB, SCSI UW
Adaptec 2940 UW
Kernel 2.2.7 se záplatami pro RAID

Tři U2W disky jsou zapojeny na řadič U2W a disky UW jsou zapojeny na řadič UW.Zdá se, že na uvedeném systému není možné protlačit přes SCSI sběrnice více než 30 MB/s ať užs použitím RAID nebo bez. Můj odhad je, že systém je příliš starý, propustnost pamětí je nic moca to limituje množství dat, která mohou přes SCSI řadič projít.

RAID-0

Čtení je *Sekvenční vstupní blok* a *Zápis* je *Sekvenční výstupní blok.* Velikost souboru je ve všech testech 1 GB. Testy byly prováděny v jednouživatelském režimu. Ovladač SCSI řadiče byl nasta-ven, aby nepoužíval *tagged command queuing* (značkovanou frontu příkazů – jedná se o vlast-nost disku a řadiče).
Chunk size označuje velikost bloku pole, *Block size* označuje velikost bloku souborového systému.

Chunk size Block size Čtení kB/s Zápis kB/s

4k 1k 19 712 18 035 4k 4k 34 048 27 061 8k 1k 19 301 18 091 8k 4k 33 920 27 118 16k 1k 19 330 18 179 16k 2k 28 161 23 682 16k 4k 33 990 27 229 32k 1k 19 251 18 194 32k 4k 34 071 26 976

Podle tabulky to vypadá, že velikost chunk size nehraje příliš velkou roli. Nicméně velikost bloku souborového systému ext2 by měla být co největší, což je na architektuře IA-32 velikost 4 kB (respektive velikost stránky).

RAID-0 s TCQ

Tentokrát byl ovladač SCSI nastaven tak, aby používal *tagged command queuing* s hloubkou fron-ty 8. Jinak vše zůstalo stejné.

Chunk size Block size Čtení kB/s Zápis kB/s

32k 4k 33 617 27 215

Více testů jsem nedělal. S TCQ se mírně zvýšila rychlost zápisu, ale ve skutečnosti o moc velký rozdíl nejde.

RAID-5

Pole bylo nastaveno do režimu RAID-5 a byly spuštěny stejné testy.

Chunk size Block size Čtení kB/s Zápis kB/s

8k 1k 11 090 6 874 8k 4k 13 474 12 229 32k 1k 11 442 8 291 32k 2k 16 089 10 926 32k 4k 18 724 12 627

Nyní již má na výkonnost vliv jak velikost bloku RAID, tak velikost bloku souborového systému.

RAID-10

RAID-10 jsou „zrcadlené stripy“ nebo RAID-1 vytvořený nad dvěma poli RAID-0. Parametr *chunk-size* je velikost bloku jak pro RAID-1, tak pro obě pole RAID-0. Netestoval jsem případ, kde by se velikosti bloků lišily, ačkoli by to mělo být bezproblémové a validní nastavení.

Chunk size Block size Čtení kB/s Zápis kB/s

32k 1k 13 753 11 580 32k 4k 23 432 22 249

Více testů nebylo uděláno. Velikost souboru byla jen 900 MB, protože čtyři oddíly, které jsem používal, měly 500 MB každý. Tím nezbývalo dost místa pro soubor velký 1GB (RAID-1 na dvou 1000MB polích).

Čerstvé testovací nástroje

Pro zjišťování rychlosti a výkonnosti vašeho RAID systému NEPOUŽÍVEJTE program hdparm.Nezjistí vám skutečnou rychlost pole.
Místo programu hdparm se podívejte na níže popsané nástroje: IOzone a Bonnie++.IOzone (*http://www.iozone.org/*) je malý a všestranný moderní nástroj. Měří výkonnost vstup-ně/výstupních operací read, write, re-read, re-write, read backwards, read strided,fread, fwrite, random read, pread, mmap, aio_read a aio_write. Nedělejte si starosti sesouborovým systémem. Může běžet na

ext2, ext3, reiserfs, JFS nebo souborovém systému XFSs OSDL STP.

Můžete také použít *IOzone* pro měření propustnosti jako funkci počtu procesů a počtu disků pou-
žitých na souborovém systému, což je zajímavé, pokud jde o stripovaný RAID.I když je dokumentace pro IOzone dostupná ve formátech Acrobat/PDF, PostScript, nroff a MSWord, ukážeme si jeden pěkný příklad IOzone v akci:

iozone -s 4096

Příkaz spustí test s použitím velikosti souboru 4 096 kB.A zde je ukázka výstupu, který IOzone generuje (výstup byl pro potřeby sazby upraven):

File size set to 4096 KB
Output is in Kbytes/sec
Time Resolution = 0.000001 seconds.
Processor cache size set to 1024 Kbytes.
Processor cache line size set to 32 bytes.
File stride size set to 17 * record size.

KB ‘reclen’ ‘write’ ‘rewrite’ ‘read’ ‘reread’ ‘random read’4096 4 99028 194722 285873 298063 265560

‘random write’ ‘bkwd read’ ‘record rewrite’ ‘stride read’ ‘fwrite’170737 398600 436346 380952 91651

‘frewrite’ ‘fread’ ‘freread’127212 288309 292633

Musíte hlavně vědět o vlastnosti, která dělá nástroj IOzone tak užitečným pro měření RAID systé-mů. Operace zohledňující RAID se jmenuje read strided. Výše uvedený příklad ukazuje, že rychlost je 380 952 kb/s. pro operaci read strided. Sami si můžete odvodit, co to znamená.

Bonnie++ (*http://www.coker.com.au/bonnie++/*) je více zaměřen na měření rychlosti jednotlivých disků než RAID systému, ale může testovat soubory větší než 2 Gb na 32bitových strojích a také můžete testovat operace creat, stat a unlink.

Další nástroje

Jsou samozřejmě vyvíjeny (i když nejsou v návodu popsány) i další nástroje pro práci se softwarovým Raidem.

Změna typu a velikosti pole RAID

Není snadné přidat další disk do existujícího pole. Nástroj na takové operace byl nedávno vyvi-nut a je dostupný na adrese *http://unthought.net/raidreconf*. Nástroj vám umožní změnit úroveň Raidu, například můžete udělat ze dvou-diskových polí RAID-1 jedno čtyř-diskové pole RAID-5. Umožní vám také změnit velikost bloku (chunk-size) a jednoduše přidávat disky.

Je ale dobré říct, že nástroj není připravený pro „produkční prostředí“. Sice vypadá, že funguje tak, jak má, ale jedná se o dlouhotrvající proces, který vám v případě problémů zaručí, že vaše data budou neobnovitelně rozházená po všech discích. Je naprosto *nezbytné*, abyste měli vytvoře-né kvalitní zálohy dříve, než se začnete pouštět do experimentování s tímto nástrojem.

Zálohy

Pamatujte si, že RAID není náhrada za dobré zálohování. Žádné množství redundance v nastavení Raidu vám nevrátí týden nebo měsíc stará data, stejně tak jako RAID nevydrží požár, zemětřesení nebo jiné pohromy.

Je vaší povinností chránit svá data, a to nejen pomocí pole RAID, ale také *pravidelnými* dobrými zálohami. Výborný nástroj pro takové zálohy je například zálohovací systém Amanda (*http://www.amanda.org/*).

Rozdělování oddílů RAID/LVM na poli RAID

Na zařízeních typu RAID není možné vytvářet oddíly jako na obyčejných discích. To může být velmi nepříjemné u systémů, kde máte například 2 disky v jednom poli RAID-1, ale chcete rozdělit systém na více nezávislých souborových systémů. Hororový příklad může vypadat třeba následovně:

df -h Filesystem Size Used Avail Use% Mounted on /dev/md2 3.8G 640M 3.0G 18% / /dev/md1 97M 11M 81M 12% /boot /dev/md5 3.8G 1.1G 2.5G 30% /usr /dev/md6 9.6G 8.5G 722M 93% /var/www /dev/md7 3.8G 951M 2.7G 26% /var/lib /dev/md8 3.8G 38M 3.6G 1% /var/spool /dev/md9 1.9G 231M 1.5G 13% /tmp /dev/md10 8.7G 329M 7.9G 4% /var/www/html

Rozdělování oddílů na zařízení typu RAID

Kdyby bylo možné na RAID zařízení rozdělovat oddíly, pak by mohl administrátor jednoduše vytvořit jedno zařízení /dev/md0 a

obvyklým způsobem jej rozdělit a vytvořit na něm souborové systémy. Místo toho ale u dnešního softwarového Raidu musí vytvořit vlastní zařízení RAID-1 pro každý souborový systém i v případě, že jsou v systému pouze 2 disky.

Je k dispozici několik různých úprav jádra, které umožňují rozdělování oddílů na RAID zařízení, ale žádná z nich (v době psaní) to nedotáhla až k zařazení do samotného jádra. Jednoduše řečeno, v současné době není možné vytvářet oddíly na zařízení typu RAID. Ale naštěstí *existuje* jiné řešení tohoto problému.

LVM na poli RAID

Řešením problému s rozdělováním oddílu je LVM – Logical Volume Management (Správa logic-kých oddílů). LVM je ve stabilních jádrech již velmi dlouho. LVM2 v jádře řady 2.6 je ještě více vylepšené oproti starší LVM podpoře v řadě 2.4. LVM tradičně odstrašuje některé lidi kvůli své komplexnosti. Nicméně jedná se opravdu o něco, co by administrátor měl zvážit, pokud chce mít na serveru více než jen pár souborových systémů.

Nebudu se zde snažit popisovat nastavení LVM, protože návod přesně s tímto zaměřením již exi-stuje. Ale malý příklad konfigurace RAID + LVM si zde ukážeme. Podívejte se na výstup progra-mu df ze stejného systému:

```
# df -h Filesystem Size Used Avail Use% Mounted on /dev/md0 942M 419M 475M 47% / /dev/vg0/backup 40G 1.3M 39G
1% /backup /dev/vg0/amdata 496M 237M 233M 51% /var/lib/amanda /dev/vg0/mirror 62G 56G 2.9G 96% /mnt/mirror /dev/vg0/webroot 97M
6.5M 85M 8% /var/www /dev/vg0/local 2.0G 458M 1.4G 24% /usr/local /dev/vg0/netswap 3.0G 2.1G 1019M 67% /mnt/netswap
```

Hlavní rozdíl spočívá v tom, že systém má pouze dvě zařízení typu RAID-1. Jeden jako hlavní sou-borový systém a jeden, který není vidět z výstupu programu df z toho důvodu, že je použit jako „fyzické zařízení“ pro LVM. Což znamená, že zařízení /dev/md1 se chová jako „úložiště“ pro všechny „svazky“ ve „skupině svazků“ pojmenované vg0.

Všechna terminologie ohledně „svazků“ je vysvětlena v návodu k LVM. Nemusíte si dělat starosti, pokud úplně nerozumíte výše popsanému. Detaily nejsou momentálně příliš důležité. Jestli bude-te chtít LVM použít, tak si stejně budete muset nejdříve příslušný návod k LVM přečíst. Co je ale důležité, jsou výhody takového nastavení oproti konfiguraci s mnoha zařízeními typu md*:*

Není zapotřebí restartovat systém, když chcete přidat nový souborový systém. To by jinak bylo nutné, protože jádro nedokáže znovu načíst tabulku oddílů z disku, který obsahuje hlavní souborový systém. A znovu vytvoření oddílů by bylo nutné pro vytvoření nového RAID zařízení pro nový souborový systém.
Změna velikosti souborového systému. LVM podporuje rychlou změnu velikosti svazku (s RAID zařízením se jedná o náročnou a dlouhotrvající operaci). Pokud máte LVM na zaří-zení RAID, tak vše, co potřebujete pro změnu velikosti souborového systému, je změna velikosti svazku, a ne samotného zařízení RAID. Se souborovým systémem typu XFS nemu-síte dokonce pro změnu jeho velikosti ani oddíl odpojovat. Ext3 (v době psaní) nepodpo-ruje změnu velikosti připojeného souborového systému, ale můžete ji změnit bez nutnos-ti restartovat systém. Jen budete muset oddíl nejdříve odpojit.
Přidání nového disku. Potřebuje více místa? Jednoduché! Prostě do systému vložte dva nové disky, vytvořte na nich RAID-1, zařaďte zařízení /dev/md2 jako nový fyzický svazek a přidejte jej do vaší skupiny svazků. To je celé! Teď máte další volné místo ve skupině svazků pro zvětšování stávajících svazků nebo přidání nových.

Jednoduše řečeno – pro servery s více souborovými systémy je LVM (a LVM2) rozhodně *velmi jed-noduché* řešení, které byste měli pro softwarový RAID zvážit. Přečtěte si návod k LVM, jestli se o něm chcete dozvědět něco více.

Úplné zálohování a obnovení systému Linux

Úvod

Představte si, že se váš pevný disk právě změnil na předražený hokejový puk. Mohlo u vás také hořet a skříň vašeho počítače nyní připomíná objekt z obrazu Salvadora Dalího. Co teď? Celkové obnovení, které se někdy označuje jako obnovení od základů, znamená obnovení počí-tačového systému po katastrofálním selhání. Chcete-li provést celkové obnovení, musíte mít úplné zálohy – nikoli pouze zálohy souborového systému, ale i informace o oddílech a další data. V

tomto návodu naleznete podrobný postup zálohování počítače se systémem Linux, který umož-ní obnovení od základů, a postup samotného obnovení.

Proces obnovení od základů lze obvykle popsat takto: Instalujte operační systém z disků produktu. Instalujte zálohovací software, abyste mohli obnovit svá data. Obnovte data. Pak můžete zkontrolovat konfigurační soubory, oprávnění atd., abyste dosáhli původního nastavení.

Tento návod poskytuje postupy a skripty, které vám ušetří opakovanou instalaci operačního systé-mu. Popsaný proces zajistí pouze obnovení souborů, které byly zálohovány z produkčního počí-tače. Obnovený systém bude mít identickou konfiguraci, což může ušetřit hodiny strávené kon-trolou konfigurace a dat. Uvedení skriptů v knize má spíše studijní charakter, pro praktickou apli-kaci doporučuji stáhnout hotové skripty z webových stránek.

Zřeknutí se odpovědnosti

Autor – Charles Curley, Linux Documentation Project – ani jiné osoby nepřijímají žádnou odpovědnost za obsah tohoto dokumentu. Použití koncepcí, příkladů a jiných obsažených informací je na vaše vlastní riziko. Dokument může obsahovat chyby a nepřesnosti, které mohou způsobit
poškození vašeho systému. Postupujte opatrně. Ačkoli je výskyt chyb velmi nepravděpodobný,

autor za ně nemůže převzít žádnou odpovědnost.

Uvedení konkrétních produktů nebo značek nelze považovat za jejich doporučení.Rozhodně lze doporučit, abyste provedli zálohu svého systému před rozsáhlejší instalací a zálo-hovali v pravidelných intervalech. Při zkoumání materiálů v tomto návodu (zejména skriptů) jetaké velmi vhodné použít testovací počítač, který můžete obětovat.

Nové verze

Tento dokument naleznete na jeho domovské stránce (*http://www.charlescurley.com/Linux-Complete-Backup-and-Recovery-HOWTO.html*) nebo na webu projektu Linux Documentation Pro-ject v mnoha formátech.

Budu velmi rád, když mi pošlete své názory na tento dokument. Bez vašich oprav, námětů a dal-ších příspěvků by tento dokument neexistoval. Své dodatky, poznámky anebo kritiku mi posílej-te na adresu: (*charlescurley@charlescurley.com*).

Přehled

Dále popsaný postup není snadný a opět může představovat riziko pro vaše data. Než jej budete potřebovat, vyzkoušejte si jej! Udělejte to jako já a *pracujte s testovacím počítačem*! První testovací – říkejme mu *cílový* – počítač pro tento návod obsahoval procesor Pentium. Na jednom pevném disku IDE byla původně nainstalována distribuce Red Hat 7.1 Linux ve verzi pro servery nebo pracovní stanice. Od té doby jsem používal několik počítačů a postupně jsem aktu-alizoval na distribuci Red Hat 8.0 a Fedora Core 1, 3 a 4. Cílový počítač neobsahuje mnoho dat, protože jsem jej nastavil jako testovací stroj, který lze obětovat. Jinými slovy jsem nechtěl tento postup zkoušet na produkčním počítači a reálných datech. Před začátkem testování jsem také pro-vedl čerstvou instalaci, abych se v případě potřeby mohl vždy vrátit ke známé konfiguraci pomo-cí opakované instalace.

> Poznámka
>
> Ukázkové příkazy většinou zobrazují, co jsem zadával při obnovení svého cílového systé mu. Možná budete používat podobné příkazy, ale s odlišnými parametry. Je na vás, abys te se drželi svého nastavení a nikoli nastavení mého testovacího počítače.

Základní postup je k dispozici v knize W. Curtise Prestona *Unix Backup & Recovery*, O'Reilly & Associates, 1999, kterou jsem příznivě hodnotil ve svém článku v časopise *Linux Journal*. Kniha je však poněkud skoupá v odpovědích na detailní praktické otázky. Které konkrétní soubory se například mají zálohovat? Která metadata byste měli uchovat a jak? Těmito otázkami se zabývá tento dokument.

Než začnete postupovat podle tohoto návodu, musíte zálohovat svůj systém běžným zálohovacím nástrojem, jako je Amanda, BRU™, tar, ArkeiaÆ nebo cpio. Poté je nutné odpovědět na otázku, jak se dostat od zničeného hardwaru do fáze, kdy lze spustit nástroj pro obnovení původních dat.

Uživatelé distribucí systému Linux založených na správci RPM (Red Hat Package Manager) by měli v rámci svých běžných záloh

ukládat také metadata RPM. Následuje jeden ze skriptů uvedených dále v návodu:
bash# rpm -Va | sort +2 -t ‘ ‘ | uniq > /etc/rpmVa.txt

Umožňuje porovnání stavu systému (přesněji stavu souborů nainstalovaných z balíčků RPM) po obnovení od základů.
Chcete-li se dostat do této fáze, musíte splnit následující požadavky:

Hardware musí znovu fungovat (po případné výměně komponent). Systém BIOS by měl být správně nastaven, včetně času, data a parametrů pevného disku. Aktuálně nelze použít odlišný pevný disk.
Jednotka IomegaÆ ZIPÆ pro paralelní port nebo odpovídající zařízení. Budete potřebovat alespoň 30 MB volného místa.
Velikost instalace moderního systému Linux s několika instalovanými jádry může přesahovat 300 MB.
Záložní média.
Minimální verze (instalace) systému Linux, která umožní spustit software pro obnovení. Tento systém bude dále označován jako „Linux pro obnovení“.

Abyste se dostali do tohoto stavu, potřebujete alespoň dva stupně záloh, možná i tři. Co přesně bude součástí záloh a ve které fázi budete zálohu vytvářet, záleží na procesu obnovení. Pokud například obnovujete server pro zálohování na pásky, pravděpodobně nebudete během obnove-ní potřebovat připojení k síti. Připojení k síti tedy zálohujte pouze v rámci pravidelných záloh.

Obnovení budete také provádět v postupných krocích. V první fázi vytvoříte oddíly, souborové systémy atd. a obnovíte minimum souborů z disku ZIP. Cílem této fáze je, abyste mohli spustit funkční počítač se síťovým připojením, páskovými jednotkami, programy pro obnovení nebo libo-volnými komponentami, které budete potřebovat v druhé fázi.

Druhá fáze (je-li vyžadována) spočívá v obnovení zálohovacího softwaru a příslušných databází. Předpokládejte například, že používáte program Arkeia a vytváříte disk ZIP pro obnovení svého zálohovacího serveru od základů. Program Arkeia udržuje na pevných discích serveru rozsáhlou databázi. Chcete-li, můžete tuto databázi obnovit z pásek. Místo toho můžete celý adresář progra-mu Arkeia (v /usr/knox) archivovat pomocí nástrojů tar a gzip a uložit tato data do jiného počí-tače pomocí připojení nfs nebo ssh. První fáze, jak je definována dále, nezahrnuje rozhraní X Win-dow. Chcete-li tedy kromě svého zálohovacího programu zálohovat i rozhraní X Window, budete se muset pustit do experimentování. Některé programy pro obnovení vyžadují rozhraní X Window.

Jestliže používáte jiný zálohovací program, je možné, že budete muset zjistit jeho požadavky. Bude nutné zjistit, které adresáře a soubory ke svému spuštění vyžaduje. Pokud pro své nástroje na zálo-hování a obnovení používáte programy tar, gzip, cpio, mt nebo dd, uložíte je na příslušný disk ZIP a obnovíte je z něj v rámci první fáze postupu, jejíž popis následuje.

Poslední fáze zahrnuje celkové obnovení z pásky nebo jiného média. Po dokončení této posled-ní fáze by mělo být možné spustit plně obnovený a funkční systém.

Omezení

Tento návod se omezuje na vytvoření minimální zálohy takové, že po jejím obnovení na nový hardware („obnovení od základů“) můžete pomocí svých běžných záloh obnovit plně funkční systém. Tento návod se vůbec nezabývá běžnými zálohami.

Přes toto úzké vymezení není tento dokument vyčerpávající. Nevyhnete se zkoumání, úpravám skriptů a testování.
Uvedené skripty obnovují data oddílů přesně tak, jak se nacházely na zdrojovém pevném disku.
To je užitečné, pokud obnovujete do shodného počítače nebo alespoň na identický pevný disk,

což ale často neplatí. Zatím lze uvést dvě řešení (která budou srozumitelnější po přečtení zbytku tohoto dokumentu):

Upravte vstupní soubor tabulky oddílů. Tuto operaci jsem prováděl už několikrát. Tímto způsobem lze také přidat nové oddíly nebo odstranit stávající (upravte však také skripty, které používají vstupní soubor tabulky oddílů).
Ručně vytvořte novou tabulku oddílů a vycházejte z ní. To je jeden z důvodů, proč skript restore.metadata nevolá skript pro obnovení oddílů (tedy rozdělení) pevného disku. Použijte skript pro obnovení.

Uvedené skripty pracují pouze se souborovými systémy ext2fs, FAT12, FAT16 a FAT32. Dokud některý aktivní dobrovolník nerozšíří tyto skripty o příslušný kód, budete k zálohování a obno-vení jiných souborových systémů potřebovat další nástroje. Vhodným kandidátem je program Par-tition Image (*http://www.partimage.org/*).

Příprava

Varování

Vytvářejte své běžné zálohy podle pravidelného rozvrhu. Pokud to nebudete dělat, je pro

vás tento návod zbytečný.

Vytvořte si záchranný disk. Já nyní používám distribuci Knoppix. Poznámky k systému Knoppix naleznete dále. Knoppix však má jeden problém: Chybí mu podpora LVM. Potřebujete-li obnovit logické svazky, zvolte distribuci, která je podporuje. K tomuto účelu se hodí distribuce finnix.

V minulosti jsem používal program tomsrbt (*http://www.toms.net/rb/*). Má kvalitní dokumentaci a dovoluje na jednu disketu umístit mnoho užitečných nástrojů. Změny skriptů, které jsem musel udě-lat kvůli kompatibilitě s novějšími systémy Linux, však bohužel způsobují problémy týkající se pro-gramu tomsrbt. Balíček tomsrbt 2.0.103 je založen na programu busybox (*http://busybox.net/*), takže příslušné poznámky mohou platit i pro jiné distribuce systému Linux, které busybox obsahují.
Libovolný použitý systém Linux bude označován jako „Linux pro obnovení“. Dále zjistěte, jak vytvořit požadovanou zálohu operačního systému, kterou budete potřebovat k obnovení svých běžných záloh. Postupoval jsem podle rad z Prestonovy knihy a použil jsem jed-notku Iomega ZIP pro paralelní port. Tyto jednotky dovolují uložit na disk asi 90 MB dat. K zálo-hování svého systému jsem potřeboval asi 85 MB, takže vám 100MB jednotka ZIP nemusí stačit.

Instalace jednotky ZIP

Instalací jednotky ZIP se zabývá dokument ZIP Drive HOWTO, který je k dispozici v rámci pro-jektu Linux Documentation Project a na své domovské stránce http://www.njtcom.com/dan-sie/zip-drive.html.

Vytvoření zálohy 1. fáze

Po vytvoření produkčních záloh je nutné uchovat informace o oddílech, abyste je mohli obnovit. Skript make.fdisk vyhledá na pevném disku data oddílů a uloží je do tří souborů. První je spus-titelný skript s názvem make.dev.x (kde „x“ je název souboru zařízení, např. hda). Druhý soubor, označený mount.dev.x, vytvoří připojovací body a připojí k nim nově vytvořené oddíly. Posled-ní soubor, dev.x, obsahuje příkazy, které potřebuje nástroj fdisk k vytvoření oddílů. Pevný disk,
pro který chcete vytvořit skripty (a tedy názvy souborů), určíte tak, že název přidruženého sou-boru zařízení předáte jako argument skriptu make.fdisk. V běžném systému IDE např. příkaz:

```
bash# make.fdisk /dev/hda
```

generuje skripty make.dev.hda, mount.dev.hda a vstupní soubor pro nástroj fdisk s názvem
dev.hda. Pokud navíc skript make.fdisk narazí na oddíl FAT, zachová spouštěcí sektor oddílu v souboru označeném dev.xy, kde x je název zařízení jednotky (např. sdc, hda) a y je číslo oddílu. Spouš-těcí sektor je první sektor oddílu s velikostí 512 bajtů. Tento sektor je obnoven současně s nově vytvořeným oddílem pomocí skriptu make.dev.hda.

Ceny pevných disků naštěstí klesají téměř tak rychle jako důvěra veřejnosti v politiky po volbách. Proto je dobré, že jsou výstupní soubory textové a lze je upravovat ručně. Jedná se o nejobtížnější, ale nejpružnější způsob, jak provést obnovu na větší náhradní disk (viz Plány).

Další metadata jsou uložena pomocí skriptu save.metadata. Skript uloží informace o oddílech do souboru fdisk.hda v kořenovém adresáři disku ZIP. Tento soubor a soubor /etc/fstab je vhod-né vytisknout, abyste měli tištěnou verzi pro případ, že někdy budete muset obnovovat data oddí-lů ručně. Chcete-li ušetřit papír, můžete se přepínat mezi dvěma virtuálními konzolami. V jedné spustíte fdisk a v druhé budete podle potřeby zpracovávat programem cat soubor /etc/fstab nebo /fdisk.hda. Přitom však hrozí riziko chyb.

Je také nutné zachovat soubory, které souvisejí s metodou obnovení. Pokud například ukládáte svá data pomocí připojení nfs, potřebujete uložit soubory hosts.allow, hosts.deny, exports atd. Jestliže dále používáte proces obnovení založený na síti, jako je např. Amanda nebo Quick Restore, neobejdete se bez uložení síťových souborů typu HOSTNAME, hosts atd. a příslušného stromu softwaru.
Nejjednodušší způsob, jak tyto a podobné problémy vyřešit, spočívá v uložení celého adresáře /etc. U moderních distribucí systému Linux neexistuje způsob, jak uložit serverovou instalaci na disk ZIP s kapacitou 100 MB. Místo zachování celé struktury je nutné zvolit mnohem selektivnější pří-stup. Které soubory jsou nutné?

Adresář /boot.
Adresář /etc a jeho podadresáře.
Adresáře potřebné při spouštění systému.
Soubory zařízení v adresáři /dev.

Chcete-li zjistit adresáře požadované při spuštění, prohlédněte si inicializační soubor spouštění /etc/rc.sysinit. Nastavuje svou cestu takto:

```
PATH=/bin:/sbin:/usr/bin:/usr/sbin export PATH
```

Metodou pokusu a omylu lze zjistit, že jsou potřebné i některé další adresáře, jako např. /dev.

V systému Linux toho bez souborů zařízení moc neuděláte.Při čtení skriptu save.metadata si všimněte, že se nemusí uložit soubory, které jsou volánypomocí absolutních cest.

Než se dostanete k funkčnímu zálohovacímu skriptu, budete možná muset projít několik cyklůzálohování, testů obnovení od základů a opakované instalace z CD. Když jsem pracoval na tomtonávodu, absolvoval jsem pět takových iterací, než jsem dosáhl úspěšného obnovení. To je jedenz důvodů, proč je nezbytné používat skripty všude, kde je to možné. Pečlivě testujte!
V systému založeném na RPM lze pomocí programu rpm určit umístění jednotlivých souborů. Chcete-li například získat úplný seznam souborů používaných balíčkem openssh, spusťte příkaz:

```
bash# rpm -ql openssh
```

Některé položky nebudete potřebovat – například manuálové stránky. Můžete projít jednu položku po druhé a rozhodnout se, zda ji zálohovat či nikoli.

Varování

Druhá fáze obnovení nepřepisuje dříve obnovené soubory. To znamená, že po úplném obnovení se použijí soubory obnovené v první fázi. Kdykoli tedy aktualizujete soubory v těchto adresářích, aktualizujte i své zálohy pro obnovení od základů!

Varování Verze nástroje tar obsažená v tomsrtbt nezachovává vlastnictví obnovených souborů. To může způsobit problémy aplikacím, jako je Amanda. Nástroj Amanda pro zálohování a obno-vení používá několik adresářů, které vlastní příslušný uživatel stejného názvu. Řešení:

Poznamenejte si, které adresáře a soubory nevlastní uživatel root.
Poznamenejte si jejich vlastníky.
Zajistěte, aby bylo v rámci procesu obnovení správně nastaveno vlastnictví. Např.:

```
bash# chown -R amanda:disk /var/lib/amanda
```

Můžete také přidat tento řádek do svých skriptů pro druhou fázi obnovení (např. do skrip-tu restore.tester).

Varování

Nástroj tomsrtbt nepodporuje obnovení vlastníků podle čísel UID/GID. Chcete-li zajistit, aby bylo možné zálohy obnovit programem tomsrtbt, odeberte z parametrů příkazového řádku nástroje tar ve funkci skriptu save.metadata parametr příkazového řádku --nume-ric-owner.

Motivy a varianty Bez jednotky ZIP

Tento proces zálohování dříve vyžadoval, abyste měli při každé záloze k dispozici jednotku ZIP. Nyní vytváří obsah disku ZIP v adresáři, který můžete zálohovat po síti. Potom stačí vytvořit disk ZIP (příkazem cp -rp) na zálohovacím serveru, když potřebujete obnovit data.

Proces zálohování bude oproti přímému zápisu na jednotku ZIP rychlejší, ale měli byste zkontro-lovat, zda se výsledný adresář na disk ZIP vejde (na základě výstupu příkazu du -hs $target.zip ve skriptu save.metadata)! Viz definice proměnné zip v daném skriptu.

U svého notebooku nemohu současně používat síťovou kartu a jednotku ZIP, takže při zálohová-ní volím tento postup. Udržuji záložní i aktuální obraz, abych se mohl vrátit k předchozí verzi, pokud počítač havaruje při zálohování.

Případně můžete na pevném disku vytvořit zálohy na několik disků ZIP a při obnovení je načíst do systému.
CD-ROM

Tato varianta se podobá možnosti bez jednotky ZIP výše. Uložte své zálohy do adresáře na pev-ném disku, jak bylo uvedeno. Potom pomocí příkazu mkisofs vytvořte z daného adresáře obraz standardu ISO 9660 a vypalte jej. Tuto možnost nelze použít s některými systémy Linux založe-nými na discích CD-ROM, jako je Knoppix, protože systém Linux vyžaduje jednotku CD-ROM. Problém lze obejít pomocí dvou jednotek CD-ROM, z nichž jedna může být například připojena přes USB. Právě z uvedeného důvodu jsem tímto způsobem připojil vypalovačku DVD.

Případně se seznamte s možnostmi přepracování distribuce Knoppix (http://www.knoppix.net/ wiki/Knoppix_Remastering_Howto) v případě záloh první a druhé fáze na disku CD-ROM/DVD. Mělo by také být možné přepracovat distribuci finnix (*http://www.finnix.org/Remastering_Finnix*).

V současnosti se mnoho počítačů dodává s jednotkou CD-ROM, ale bez disketové jednotky. Dis-ketové jednotky jsou navíc nespolehlivé. Proto je rozumné vypálit disk CD-ROM se spustitelným obrazem. Formát „El Torito“ je bohužel kompatibilní s

disketami velikosti 1,2 MB, 1,44 MB a 2,88 MB a program tomsrtbt pracuje s disketou velikosti 1,7 MB. Naštěstí však můžete získat verzi pro diskety 2,88 MB s názvem tomsrtbt-2.0.103.ElTorito.288.img ze stejných mirrorů, kde jste získali obraz pro normální diskety. Umístěte *kopii* [1] do kořenového adresáře se záložními soubo-ry. Potom příkazem mkisofs s parametrem -b nastavte tomsrtbt-2.0.103.ElTorito.288.img jako soubor spouštěcího obrazu.

Jediná nevýhoda tohoto procesu spočívá v tom, že mnoho starších systémů BIOS nepodporuje obrazy disket velikosti 2,88 MB na discích CD-ROM. Většina z nich umožní spuštění z diskety programu tomsrtbt.

Alternativou je použít Syslinux (*http://syslinux.zytor.com/*). Nezávisí na obrazu diskety a umožňu-
je vytvořit vlastní disk CD s mnoha nástroji (např. programem tomsrtbt).Chcete-li počítač spustit z jednotky CD-ROM, je někdy nutné změnit nastavení systému BIOS. Jest-liže to nelze provést, buď kvůli tomu, že systém BIOS nepodporuje spouštění z jednotky CD-ROM,nebo proto, že se nelze do systému BIOS dostat, prozkoumejte možnosti programu Smart BootManager (SBM) (*http://btmgr.webframe.org/*), uvedeného v kapitole „Zdroje informací“.

Vyzkoušejte své disky CD v jednotce, kterou budete používat při obnovení. Potřebujete-li skripty přizpůsobit, můžete je zkopírovat do adresáře /tmp, což je RAM disk vytvořený programem tomsrtbt, a upravit je v tomto umístění. Skripty lze odtamtud i spustit. RAM disk neumožňuje trva-lé uložení, takže nezapomeňte své změny před restartem počítače uložit!

Více disků ZIP

Rozdělíte-li dva skripty první fáze restore.metadata a save.metadata, můžete rozložit meta-data první fáze na více disků ZIP.

Vyloučení z první fáze ukládání

V některých případech je nutné omezit data první fáze o několik megabajtů, zejména pokud se snažíte vejít do limitu disku ZIP. Funkce crunch ve skriptu save.metadata přijímá více para-metrů, které předává programu tar. Akceptuje také parametr --exclude. Můžete tedy například vyloučit adresáře samba a X11 v adresáři /etc takto:

```
crunch etc --exclude etc/samba --exclude etc/X11 etc
```

Proč zrovna tyto dva adresáře? Spotřebují totiž hodně místa na pevném disku a při spouštění nejsou potřeba.
Pokud udržujete více jader, můžete vyloučit moduly pro všechna jádra, která nebudete spouštět. Zkontrolujte soubor lilo.conf nebo grub.conf a podívejte se, které jádro budete používat. Pak v souboru /lib/modules vyhledejte adresáře modulů, které lze vyloučit.

Jak najít další vhodné kandidáty na vyřazení? Vypište seznam jednotlivých souborů v cílových adresářích příkazem ls -alSr a velikost adresářů příkazem du | sort -n.Další (asi elegantnější) způsob, jak vyloučit adresáře, je založen na vypsání úplného seznamuadresářů do souboru, na který se lze odkazovat pomocí možnosti --exclude-from=NAZEVSOUBORU programu tar.

Initrd

Jestliže se systém spouští s počátečním RAM diskem neboli initrd, zkontrolujte, zda skript resto-re.metadata vytváří adresář /initrd. Nejsnáze to můžete zajistit tak, že jej zahrnete na konec seznamu adresářů, který slouží pro smyčku vytvářející adresáře.

Systém pravděpodobně používá initrd, pokud se spouští z jednotky SCSI nebo má kořen na oddí-lu ext3fs. V souboru /etc/lilo.conf zjistíte, zda obsahuje příslušné volání. Initrd dnes používá většina moderních distribucí.

> Poznámka
>
> Zdůrazňuji kopírování, protože příkaz mkisofs poškodí soubor v adresáři, ze kterého vytváří obraz ISO.

První fáze obnovení

Spouštění

Nejdříve je nutné ověřit, zda je správně nastaven hardwarový čas. K tomu slouží nastavení systé-mu BIOS. Požadovaná přesnost nastavení času závisí na používaných aplikacích. Při obnovení by měla stačit odchylka několika minut od správného času. Když nakonec spustíte obnovený systém, mohou díky tomu časově kritické události pokračovat ze stavu, ve kterém byly zastaveny.

tomsrtbt

Před spuštěním nástroje tomsrtbt zkontrolujte, zda je na paralelním portu (/dev/lp0 nebo /dev/lp1) nainstalována jednotka ZIP. Spouštěcí software načte ovladač jednotky ZIP pro para-lelní port automaticky.

Další krok spočívá v nastavení grafického režimu. Osobně toho chci na obrazovce vidět co nejví-ce. V nastavení grafického režimu volím režim 6 s 80 sloupci a 60 řádky. Váš hardware nemusí takto vysoké rozlišení zvládnout, takže je nutné dostupné možnosti vyzkoušet.

Knoppix

Tyto pokyny budou pravděpodobně platit i pro jiné systémy Linux na disku CD-ROM nebo USB flash disku, ale v některých bodech se mohou lišit. Před spuštěním distribuce Knoppix zkontrolujte, zda je na paralelním portu /dev/lp0 nebo /dev/lp1 nainstalována jednotka ZIP nebo odpovídající zařízení. Systém Knoppix automaticky nenačte ovladač jednotky ZIP pro paralelní port. Místo toho jej instalujte příkazem modprobe ppa (jako uživatel root).
Spusťte Knoppix standardním způsobem. Podle mých zkušeností je rychlejší a užitečnější spustit systém v konzolovém (textovém) režimu. Ze spouštěcí nabídky vyberte příkaz „knoppix 2“. Pak se nastavte jako uživatel root příkazem su -. Místo hesla pouze stiskněte klávesu Enter.

Finnix

Jednou z možností spouštění systému finnix je „toram“. Umožní přesunout veškerý kód do pamě-ti RAM. Díky tomu lze vložit do jednotky další disk CD s daty první fáze.

Obnovení

Tyto pokyny předpokládají, že pracujete s nástrojem tomsrtbt. Pokud ve svém systému pro obno-vení používáte jinou verzi systému Linux, nejspíš budete muset tyto pokyny poněkud upravit. Tyto skripty byste například vždy měli spouštět jako uživatel root, i když vám některý jiný uživatel poskytuje požadovaná oprávnění.

Po spuštění systému Linux pro obnovení a po zobrazení konzoly připojte jednotku ZIP. Pravděpodobně je vhodné ji připojit pouze pro čtení:

```
# mount /dev/sda1 /mnt -o ro
```

Zkontrolujte, zda je k dispozici:

```
# ls -l /mnt
```

V systému Knoppix nebo finnix můžete vytvořit adresář pod adresářem /mnt a připojit jednotku v tomto bodě, např. takto:

```
# mkdir /mnt/zip # mount /dev/sda1 /mnt/zip -o ro
```

V této fázi můžete spustit obnovení automaticky nebo ručně. Automatické obnovení zvolte v pří padě, že během celé operace nepotřebujete provádět žádné změny. Jedním z hledisek je přítomnost více pevných disků v počítači. Pokud vaše instalace systému Linux připojuje oddíly na více pevných discích, musíte nejdříve připojit kořenový oddíl. Tím zajistíte, že adresáře připojovacích bodů vzniknou na správném oddílu. Skript first.stage spustí skripty pro připojení jednotek v pořadí, ve kterém jsou vytvořeny. Jestliže jste je vytvořili (ve skriptu save.metadata) v pořadí, ve kterém se větví z kořenového adresáře, měl by proces připojování fungovat hladce.

Máte-li více pevných disků a připojování probíhá křížově, musíte si poradit sami. Můžete skripty buď zkombinovat a upravit, aby připojení oddílů proběhlo ve správném pořadí, nebo je připojit ručně.

Automaticky

Automatický proces zavolá všechny ruční skripty ve správném pořadí. Neumožňuje ruční zásah, řekněme při vytvoření systémů souborů, které tento návod nepodporuje. Chcete-li spustit první fázi obnovení automaticky, zadejte příkaz:

```
# /mnt/root.bin/first.stage
```

Pokud chcete zkontrolovat chybné bloky, doplňte parametr -c.

Ručně

Jestliže chcete proces spustit ručně, přejděte do adresáře na jednotce ZIP, kde jsou umístěny skripty.

```
# cd /mnt/root.bin
```

Nyní spusťte skript nebo skripty, které obnoví informace o oddílech a vytvoří souborové systémy. Můžete je spustit v libovolném pořadí. Např.:

```
# ./make.dev.hda
```

Pokud chcete zkontrolovat chybné bloky, doplňte parametr -c. Funkce tohoto skriptu:

Vyčistí prvních 1 024 bajtů pevného disku a tím odstraní jakoukoli existující tabulku oddí-lů a hlavní spouštěcí záznam (MBR).
Znovu vytvoří oddíly podle informací získaných při spuštění skriptu make.fdisk.
Vytvoří požadované oddíly se souborovými systémy ext2 a ext3 a odkládací oddíly syste-mu Linux. Zahrnete-li do skriptu parametr -c, proběhne také kontrola chybných bloků.
Vytvoří některé typy oddílů FAT.

Nyní je správný čas na kontrolu geometrie jednotky. Různé verze systému Linux někdy volí odliš-né geometrie, takže výchozí geometrie v souboru dev.hdX nemusí být správná. Chcete-li vynutit správnou geometrii v systému Knoppix, upravte skript make.dev.x. Pomocí parametrů -C, -H anebo -S programu fdisk určete počet cylindrů, hlav a sektorů. Tyto hodnoty lze získat ze sou-boru fdisk.hdX v kořenovém adresáři jednotky ZIP. Pak spusťte skript znovu.

> Poznámka
>
> Pokud chcete obnovit jiné operační systémy nebo souborové systémy, je vhodné to provést v této fázi. Po dokončení operace restartujte systém Linux pro obnovení a pokračujte v obnově dat.

Nyní spusťte skript nebo skripty, které vytvoří připojovací body a připojí k nim oddíly.

```
# ./mount.dev.hda
```

Jakmile vytvoříte všechny adresáře a připojíte k nim oddíly, můžete spustit skript restore.meta-data.

```
# ./restore.metadata
```

Tento skript obnoví obsah jednotky ZIP na pevný disk.Měli byste najít adresář s kořenovým adresářem disku ZIP a pak seznam obnovených archivníchsouborů. Po zadání příkazu tar v nástroji tomsrtbt zjistíte, že velikost bloku programu tar je 20,což je v pořádku. Tuto zprávu můžete ignorovat. Zkontrolujte, zda zavaděč LILO (příkaz lilo)vypsal své výsledky:

```
Added linux *
```

Dále bude následovat výstup příkazu „df -m“.

Závěrečné úpravy

Pokud obvykle spouštíte přímo systém X Window, můžete se setkat s určitými problémy. Pro jis

totu dočasně změňte svou úroveň spouštění.Spouštíte-li pomocí zavaděče grub, vyberte v okně výběru grub jádro, které chcete spustit. Stisk-nutím klávesy e přejděte do režimu úprav a přidejte k řádku jádra mezeru a číslici „3“. Potvrďtenový řádek a spusťte systém stisknutím klávesy b.
Jestliže nepoužíváte grub, upravte před spuštěním soubor /target/etc/inittab. Vyhledejte řádek následujícího tvaru:

```
id:5:initdefault:
```

a změňte jej na:

```
id:3:initdefault:
```

Nyní můžete bezpečně restartovat. Pokud jste to zatím neudělali, vyjměte ze spouštěcí jednotky médium. Nyní můžete bezpečně restartovat:

```
# shutdown -r now
```

nebo

```
# reboot
```

Počítač se restartuje.

Druhá fáze obnovení

Při restartu počítače přejděte znovu do systému BIOS a zkontrolujte, zda hodiny ukazují přibližně
správný čas. Po ověření hodin ukončete systém BIOS a restartujte z pevného disku. Můžete jednoduše pone-chat normální sekvenci spouštění systému. Zobrazí se mnoho chybových zpráv, většinou násle-dujícího typu: „Nelze najít XY!“ Pokud jste zatím postupovali správně, na těchto chybových zprá-vách nezáleží. Požadované operace zvládnete i bez programů linuxconf či apache.

> Poznámka
>
> Případně můžete spustit systém v jednouživatelském režimu (na výzvu zavaděče lilo zadej-te linux single), ale v tomto případě bude nutné konfigurovat síť ručně a spustit démo-ny potřebné k obnovení systému (např. sshd). Příslušný postup hodně záleží na systému.

Mělo by být možné se přihlásit ke konzole uživatele root (bohužel chybí systém X Window i uži
vatelé). Nyní by měla být k dispozici síť, například k připojení záloh systému pomocí nfs. Pokud jste vytvořili navrženou

dvoufázovou zálohu pro program Arkeia, můžete nyní obnovit databázi a spustitelné soubory tohoto programu. Měl by fungovat příkaz:

/etc/rc.d/init.d/arkeia start

pro spuštění serveru. Je-li v jiném počítači se systémem X Window nainstalováno grafické uživa-telské rozhraní, můžete se nyní přihlásit k programu Arkeia na serveru pro zálohování na pásky a připravit obnovení dat.

> Poznámka
>
> Při obnovování si pečlivě prostudujte dokumentaci k programům pro obnovení. Program tar například normálně neobnovuje některé vlastnosti souborů, jako jsou bity SUID. Oprávnění k souborům se určují podle hodnoty umask uživatele. Chcete-li obnovit sou-bory přesně tak, jak jste je uložili, zvolte přepínač -p programu tar. Analogicky tedy zkon-trolujte, zda software pro obnovení obnovuje všechna data v původní podobě.

Pokud chcete obnovit testovací počítač, zadejte:
bash# restore.all

Jestliže jste zálohovali a obnovovali data pomocí programu tar a zadali jste parametr -k (ponechat a nepřepisovat starší soubory), zobrazí se mnoho následujících zpráv:

tar: usr/sbin/rpcinfo: Could not create file: File exists tar: usr/sbin/zdump: Could not create file: File exists tar: usr/sbin/zic: Could not create file: File exists tar: usr/sbin/ab: Could not create file: File exists

To je normální, protože program tar odmítá přepsat soubory, které jste obnovili v první fázi. Poté restartujte počítač. Při vypínání uvidíte mnoho chybových zpráv jako „no such pid“. Jedná se
o normální součást procesu. Kód pro vypnutí používá soubory pid démonů, které byly spuštěny při vytváření zálohy, a snaží se vypnout démony, které při posledním spuštění systému spuštěny nebyly. Příslušné identifikátory pid samozřejmě neexistují.

Systém by měl naběhnout normálně s mnohem menším počtem chyb než posledně. V ideálním případě by nemělo dojít k žádným chybám. V systému založeném na RPM můžete spolehlivost obnovení nejlépe zjistit, když ověříte všechny balíčky:

bash# rpm -Va | sort +2 -t ‘ ‘ | uniq > ~/foo.txt diff /etc/rpmVa.txt ~/foo.txt

Chybové zprávy prelinku jsou normální a můžete je ignorovat. Případně je můžete odstranit pří
kazem /etc/cron.daily/prelink. Při normální činnosti systému se některé soubory mění (např. konfigurační soubory a logy). Těch-to souborů si nevšímejte. Výstup lze přesměrovat do souboru a programem diff jej porovnat se souborem, který jste vytvořili v době zálohování (/etc/rpmVa.txt). Tento krok se tím značně urychlí. Uživatelé programu Emacs mohou vyzkoušet integrované možnosti pro porovnávání.

V této fázi by měl být systém funkční. Je čas vyzkoušet aplikace, zejména takové, které jsou spuš-těny jako démon. Čím komplikovanější aplikace, tím rozsáhlejší testování bývá potřeba. Máte-li vzdálené uživatele, zakažte jim pracovat se systémem nebo jej v průběhu testování nastavte pouze pro čtení. Tento krok je zvláště důležitý pro databáze, abyste zabránili tomu, že se případné poškození nebo ztráta dat ještě zhorší.

Pokud běžně spouštíte do systému X Window a v předchozích fázích jste jej vypnuli, vyzkoušej-te jej dříve, než jej znovu zapnete. Systém X Window znovu aktivujete změnou jediného řádku v souboru /etc/inittab zpět na:

id:5:initdefault:

Nyní můžete začít slavit – a pak máte nárok na odpočinek.

Poznámky specifické pro konkrétní distribuce

Následují poznámky k distribucím, které jsou založeny na předchozích zkušenostech. Pokud máte další poznámky k jiným distribucím, které byste chtěli doplnit, pošlete mi je prosím.

Fedora Core 3 a 4

Skripty jsou nyní určeny pro distribuci Fedora Core 4, takže pravděpodobně nebude nutné tyto skripty měnit.

> Poznámka
>
> Výše uvedené skripty jsem vyzkoušel v nové instalaci FC3. Setkal jsem se s problémy se zařízeními po spuštění, když jsem pracoval v systému, který byl aktualizován z FC2 na FC3.

Knoppix

Nedávno jsem začal používat distribuci Knoppix. Pasi Oja-Nisula informuje: „Pro mě je na distribuci Knoppix nejlepší, že

nepotřebuji speciální spouštěcí médium pro každý počítač, ale vždy můžu použít stejné nástroje. Systém Knoppix navíc opravdu skvěle
podporuje hardware. Nemám příliš mnoho zkušeností s odlišnými platformami, ale všech-ny počítače, které jsem vyzkoušel, fungovaly správně, systém vyhledal zařízení SCSI atd. Při tomto procesu obnovení kopíruji zálohy po síti na jiný počítač. Při obnovení spouštím
pomocí disku CD s distribucí Knoppix a načítám soubor metadata.tar.gz ze síťového počítače. Potom spouštím skripty make.dev a mount.dev, načítám další soubory typu tar.gz, nastavím grub a restartuji. Občas je potřeba něco zadat, ale díky těmto skriptům je postup docela přímočarý. Výjimku představuje přechod z IDE na SCSI apod., ale ani to není tak obtížné, protože systém Linux lze snadno obnovit na jiný hardware.“

Rád bych dodal, že systém Knoppix automaticky detekuje zařízení USB, což je skutečně milé. Tato

média představují skvělou (a prostornější) náhradu jednotky ZIP.Viz také stránka „System recovery with Knoppix“ (http://www-106.ibm.com/developerworks/linux/library/l-knopx.html?ca=dgr-lnxw04Knoppix).

Při obnovení pracujte jako uživatel „root“, nikoli jako uživatel „knoppix“. Jinak mohou být někte-ré obnovené adresáře a soubory vlastněné podivným uživatelem nebo skupinou. Při práci s distri-bucí Knoppix je také nutné archivovat pomocí programu tar data první fáze s uložením číselnýchhodnot uživatele a skupiny, nikoli podle jména. Jména mohou v systému Knoppix odkazovat najiná čísla, takže by se soubory neobnovily správně.

finnix

Finnix poskytuje podobné výhody jako distribuce Knoppix. Kromě toho se spouští v režimu pří-kazového řádku s podporou myši, což je pro účely obnovení ideální. Systém je malý, v době psaní tohoto textu zabírá méně než 100 MB, takže jej můžete přepracovat se svými daty první fáze. Spouští se rychle. Nabízí také podporu LVM. Kromě toho obsahuje Zile, což je klon Emacsu. K účelům obnovení systému jsem s distribucí finnix velmi spokojen.

Poznámky specifické pro aplikace

Následují různé poznámky o zálohování konkrétních aplikací.

Logical Volume Manager (Správce logických svazků)

Zpracování logických svazků bývá poněkud obtížné: Pomocí spouštěcího kódu distribuce finnix můžete LVM zapnout a vypnout. Z toho vyplývá, že kód první fáze obnovení bude specifický pro jednotlivé distribuce. Je generován v souboru make.fdisk. Chcete-li jej upravit, vyhledejte ve skriptu make.fdisk řetězec „Ošklivě“ (anglicky „Hideous“).

LVM vyžaduje přidání dvou nových speciálních skriptů make.lvs a mount.lvs. Generují se a pou-žijí pouze v přítomnosti logických svazků.

SElinux

SElinux je v testovacích počítačích vypnutý. Adresář /selinux se v žádném z těchto skriptů nezá-lohuje. Dá se odhadnout, že byste pravděpodobně měli SElinux vypnout po první fázi obnovení, a než jej znovu zapnete, nejspíš bude nutné provést určité s ním související úkoly.

GRUB

Výchozím zavaděčem v distribuci Fedora je GRUB (Grand Unified Bootloader). Je nutné jej spus-tit na konci první fáze, aby bylo možné následně počítač spustit. Chcete-li jej zachovat pro první fázi obnovení, proveďte následující změny:

- Upravte předposlední sekci skriptu restore.metadata:

```
# Nyní nainstalujte spouštěcí sektor.
# chroot $target /sbin/lilo -C /etc/lilo.conf
chroot $target /sbin/grub-install /dev/hda
```

- Do skriptu save.metadata přidejte následující sekci:

```
# Grub vyžaduje tyto hodnoty při instalaci.
if [ -d usr/share/grub ] ; then # Red Hat/Fedora
    crunch usr.share.grub usr/share/grub
fi
if [ -d usr/lib/grub ] ; then # SuSE
```

```
    crunch usr.lib.grub usr/lib/grub
fi
```

Tripwire

Pokud používáte Tripwire nebo libovolnou jinou aplikaci, která spravuje databázi metadat o souborech, vytvořte tuto databázi znovu ihned po obnovení.

Squid

Squid je server proxy a mezipaměť pro HTTP. Proto na pevném disku udržuje mnoho dočasných dat. Není důvod tato data zálohovat. Do příslušného příkazu tar ve svém skriptu druhé fáze zálo-hování zadejte parametr --exclude /var/spool/squid. Potom ponechejte program squid, aby adresářovou strukturu obnovil automaticky. Na konec skriptu druhé fáze obnovení přidejte příkaz, který zajistí, aby se program squid inicializoval. Já jsem přes připojení ssh do skriptu resto-re.tester vložil:

```
ssh $target "mkdir /var/spool/squid ; chown squid:squid /var/spool/squid;\ /usr/sbin/squid -z;touch /var/spool/squid/.OPB_NOBACKUP"
```

Poslední příkaz vytvoří soubor s nulovou délkou a názvem .OPB_NOBACKUP. Tento soubor je určen pro aplikaci Arkeia, které sděluje, že nemá zálohovat data pod tímto adresářem.

Arkeia

Tyto poznámky vycházejí z testování s verzí Arkeia 4.2.Arkeia (*http://www.arkeia.com/*) je program na zálohování a obnovení, který funguje na mnoha
různých platformách. Program Arkeia můžete použít jako součást schématu obnovení od základů,
ale jsou zde dvě komplikace.
První je pravděpodobně nejzávažnější – neexistuje elegantnější řešení, než při obnovení v navi

gátoru ručně vybírat adresáře, které chcete obnovit. Je to zřejmě způsobeno tím, že program Arke-ia neposkytuje žádnou funkci k obnovení souborů, které se již nacházejí na disku, což by odpo-vídalo parametru -p programu tar. Pokud jednoduše povolíte úplné obnovení, proces havaruje,protože Arkeia přepíše některou knihovnu používanou při obnovení, např. lib/libc-2.1.1.so.Ruční výběr adresářů pro obnovení je přinejmenším riskantní, takže jej nedoporučuji.
Druhý zádrhel spočívá v tom, že je nutné zálohovat datový slovník programu Arkeia. Chcete-li to provést, upravte skript save.metatdata přidáním programu Arkeia do seznamu ukládaných adre-sářů:

```
# určeno pro program arkeia: tar cf -usr/knox | gzip -c > $zip/arkeia.tar.gz
```

Datový slovník *musíte* zálohovat tímto způsobem, protože Arkeia tento slovník nezálohuje. Je to jedna z věcí, které se mi na programu Arkeia nelíbí. Problém ve svém počítači řeším tak, že uklá-dám datový slovník na pásku pomocí řešení BRU od TOLIS Group *(http://www.estinc.com/*).

Datový slovník je automaticky obnoven pomocí skriptu restore.metadata.

Amanda

Amanda (*http://www.amanda.org/* – The Advanced Maryland Automatic Network Disk Archiver) spolupracuje s touto sadou skriptů docela dobře. Zvolte v programu Amanda normální proces zálohování a vytvořte svá data první fáze jako obvykle. Amanda ukládá data na pásku ve formá-tu programů tar nebo cpio. Umožňuje obnovení v rozsahu od jednotlivých souborů po celé zálož-ní obrazy. Obnovení celých obrazů je výhodné v tom, že pak můžete pomocí variant skriptů v tomto návodu zvolit obnovení z obrazů nebo přímo z pásku. Podařilo se mi obnovit testovací počítač podle pokynů v knize W. Curtis Prestona *Unix Backup & Recovery*. Další informace o této knize naleznete v kapitole „Zdroje informací". Kapitola této knihy o programu Amanda je k dis-pozici online (*http://www.backupcentral.com/amanda.html*).

Ve skriptu restore.tester jsem udělal dvě změny. Za prvé jsem jej upravil tak, aby jako argu-ment přijímal název souboru. Protože příkaz amrestore programu Amanda dekomprimuje data během obnovení, přepsal jsem jej dále tak, aby soubor programem cat připojil do roury, místo aby jej dekomprimoval.

Výsledný řádek vypadá takto:

```
cat $file | ssh $target "umask 000 ; cd / ; tar -xpkf -"
```

kde $file je argument skriptu, tj. obraz obnovený z pásku příkazem amrestore.Vzhledem k tomu, že argumenty příkazového řádku programu tar zabraňují přepsání, obnovujte
z obrazů v *opačném* pořadí, než ve kterém byly vytvořeny. Nejaktuálnější soubory obnovte jakoprvní.Jestliže zálohujete datový adresář pomocí skriptu save.metadata, vyžaduje program Amanda

ruční nastavení vlastnictví. Příklad:

```
bash# chown -R amanda:disk /var/lib/amanda
```

Můžete také přidat tento řádek do svých skriptů pro druhou fázi obnovení (např. do skriptu res-tore.tester).

NTFS

Dobře, NTFS není aplikace. Jedná se o souborový systém používaný v operačním systému Windows NT společnosti Microsoft a jeho následnících, včetně Windows 2000 a Windows XP. Můžete jej zálo-hovat a obnovit ze systému Linux pomocí programu ntfsclone, což je jeden z nástrojů pro NTFS v sadě ntfsprogs, která je k dispozici na adrese *http://linux-ntfs.sourceforge.net/downloads.html*.

Tyto skripty vytvoří oddíly NTFS, ale neumístí na ně souborový systém. Z dokumentace není jasné, zda nástroj ntfsclone zapíše souborový systém na čistý oddíl.

Co dále?

Tento návod vznikl na základě pokusů s jedním počítačem. Nepochybně naleznete některé další adresáře nebo soubory, které budete potřebovat uložit ve své první fázi zálohování. Nezabýval jsem se uložením a obnovením systému X Window během první fáze ani jsem se vůbec nedostal k jiným procesorům než Intel.

Ocenil bych vaše reakce založené na testování a zdokonalování těchto skriptů ve vašich počíta-čích. Uvítal bych také, kdyby dodavatelé zálohovacích programů popsali, jak vytvořit minimální zálohu svých produktů. Byl bych rád, kdyby celá linuxová komunita mohla poněkud lépe spát.

Plány

Dobrovolníci jsou vítáni s otevřenou náručí. Než začnete na něčem pracovat, obraťte se na mě pro případ, že se tímto bodem již někdo zabývá.

- Chybí způsob, jak zjistit jmenovku odkládacího oddílu. To znamená, že nelze nastavit jme-novku (label) odkládacího oddílu při obnovení. Dalo by se předpokládat, že systém s jedi-ným odkládacím oddílem (podle informací programu fdisk) má jmenovku uvedenou na řádku odkládacího oddílu v souboru /etc/fstab, ale tento postup funguje pouze v systé-mech s jediným pevným diskem a mohl by způsobit záludné chyby v systémech s více odkládacími oddíly.
  Můžete to obejít tak, že přidáte jmenovku ručně opakovaným spuštěním programu mks-wap s parametrem -L. Bohužel.
- Editor oddílů pro nastavení hranic oddílů v souboru dev.hdx. Umožní uživatelům přizpůso-bit oddíly pro jiný pevný disk nebo stejný disk s odlišnou geometrií, případně upravit velikosti oddílů na stejném pevném disku. Zde by se pravděpodobně hodilo grafické uživatelské roz-hraní. Na druhou stranu se zdá, že nástroj parted (*http://www.gnu.org/software/parted*) nada-ce FSF by mohl část požadavků splňovat. Skutečně mění velikost existujících oddílů, ale má jistá omezení.
- Skript make.fdisk aktuálně rozpoznává pouze některé oddíly FAT, nikoli všechny. Při-dejte do skriptu make.fdisk kód pro detekci jiných oddílů a do výstupních souborů doplňte příslušné pokyny pro jejich obnovení.
- Oddíly FAT12 nebo FAT16 se nebudou formátovat, ale zapíší se na ně nuly, aby formáto-vání nezmátlo systém MDOS 6.x. Vysvětlení problému naleznete v poznámkách k progra-mu fdisk.
- Překlady do jiných (přirozených) jazyků.
- Občas jsem se odkazoval na správce RPM (Red Hat Package Manager). Jaké jsou odpovídající příkazy k balíčkům deb?
- Upravte kód první fáze zálohování, aby ukládal pouze aktuální jádro.

Několik rad k obnovení po haváriích

Ještě si dovolím několik rad k obnově po havárii. Disky ZIP pro jednotlivé počítače a příslušné výtisky byste měli uložit na bezpečném místě svého pracoviště. Jejich kopie je vhodné uložit do skladovacího umístění mimo pracoviště. Hlavní význam skladu záloh mimo pracoviště spočívá v tom, že umožňuje obnovení po haváriích. Součástí obnovení po haváriích je i obnovení všech hostitelských počítačů na náhradní hardware.

Ve skladu mimo pracoviště byste také měli uchovávat několik disket nebo disků CD-ROM pro obnovení systému Linux a případně i několik disků ZIP. Dále vytvořte kopie záchranných distribucí systému Linux na několika svých počítačích, aby se vzájemně zálohovaly.

Vedle záloh a ve skladu mimo pracoviště byste nejspíš měli mít i kopie tohoto návodu se svými poznámkami, které se týkají specifik vašeho hardwaru.

Skripty

Souhrn funkce jednotlivých skriptů naleznete v poznámkách na jejich začátku.

První fáze

make.fdisk

Tento skript spuštěný při zálohování vytvoří skripty podobné skriptům make.dev.hda a mount.dev.x, které můžete spustit při obnovení. Generuje také datové soubory typu dev.hda. Název skriptu a vytvořeného datového souboru závisí na zařízení, které je tomuto skriptu předá-no jako parametr. Příslušný skript spuštěný při obnovení vytvoří oddíly na pevném disku. Skript make.fdisk je volán z následujícího skriptu save.metadata.

```
#! /usr/bin/perl

# Skript v jazyku perl, který vytváří skript a vstupní soubor programu fdisk # k obnovení oddílů na pevném disku a formátování oddílu systému Linux a # odkládacích oddílů Linux. První parametr je úplná cesta # k zařízení pevného disku, např. /dev/hda. Výsledkem # jsou dva soubory: skript make.dev.x a datový soubor dev.x # (kde x je popsaný pevný disk, např. hda, sdc). Skript make.dev.x se # spouští při obnovení za účelem obnovení pevného disku x, před spuštěním # skriptu restore.metadata. dev.x je vstupní soubor programu fdisk.

# Časové razítko: <2006-04-08 15:23:55 ccurley make.fdisk>

# Copyright 2001 až do data poslední změny Charles Curley # s výjimkou subrutiny cut2fmt.

# Diskuse:

# fdisk generuje soubor v následujícím formátu, spustíte-li jej jako "fdisk # -l".

# root@tester ~/bin $ fdisk -l /dev/hda

# Disk /dev/hda: 64 heads, 63 sectors, 1023 cylinders # Units = cylinders of 4032 * 512 bytes

# Device Boot Start End Blocks Id System # /dev/hda1 1 9 18112+ 83 Linux # /dev/hda2 10 1023 2044224 5
Extended # /dev/hda5 10 368 723712+ 83 Linux # /dev/hda6 369 727 723712+ 83 Linux # /dev/hda7 728 858
264064+ 83 Linux # /dev/hda8 859 989 264064+ 83 Linux
# /dev/hda9 990 1022 66496+ 82 Linux swap


# Program fdisk neposkytuje výstup vhodný k pozdějšímu
# importu do programu fdisk ve stylu programu sfdisk. Tento skript analyzuje
# výstup z programu fdisk a vytvoří vstupní soubor pro fdisk. Vstupní soubor
# použijte takto:


# fdisk /dev/hdx < dev.hdx


# V případě sady pro obnovení od základů tento skript také generuje skript,
# který spustí výše uvedený příkaz, abyste jej mohli spustit
# z disku ZIP. Všechny skripty pro obnovení od základů jsou v adresáři /root/bin.
# Datový soubor a skript vytvořené tímto skriptem jsou proto také
# umístěny sem. Stejný skript navíc vytvoří příslušné souborové systémy pro Linux,
# bu ext2fs nebo odkládací oddíl Linux. K dispozici je omezená podpora systémů
# FAT12, FAT16 a FAT32. U libovolných jiných systémů se musíte snažit sami.


# Poznámka k systému FAT32: Podle databáze MS KB je v tomto systému více než jeden
# vyhrazený sektor - obvykle 32, ale počet může kolísat. Detaily naleznete
# v databázi M$ KB po vyhledání řetězce "boot sector" nebo BPB. Podrobnosti
# nad rámec toho, co skutečně potřebujete o použití spouštěcích sektorů, viz
# http://support.microsoft.com/support/kb/articles/Q140/4/18.asp


# Lze také změnit velikost oddílů úpravou souboru dev.x. Nezapomeňte,
# že pokud jste změnili velikost oddílu FAT přes hranici 32 MB,
# musíte také změnit typ! Spuste příkaz typu "fdisk /dev/hda"
# a pak příkaz l k zobrazení dostupných
# typů oddílů. Dále příslušným způsobem upravte soubor dev.x. Při ručním
# přesunu hranic oddílů také nezapomeňte vhodně přesunout hranice
```

```
# logických i rozšířených oddílů.


# Kontrola chybných bloků nyní spočívá v rychlém čtení oddílu. Lze
# také provést kontrolu zápisu, ale je to obtížnější. Musíte spustit
# badblocks jako samostatný příkaz a předat seznam chybných bloků
# programu mke2fs v souboru (v adresáři /tmp, což je RAM disk). Také je nutné
# znát velikost bloků, kterou zjistíte spuštěním programu
# dumpe2fs. Věci se tím komplikují a zatím jsem to nedělal. Pravděpodobně
# to není nutné pro nový pevný disk, ale pokud váš pevný disk
# havaroval a používáte jej znovu (než vám dojde
# náhradní kus, předpokládám), pak to rozhodně doporučuji. Dejte
# mi vědět, jak při tom postupujete.


# Další informace poskytne autor Charles Curley na
# http://www.charlescurley.com/.


# Program cut2fmt zjistí formátovací řetězec pro funkci rozbalení, která
# umožňuje zpracovat výstup programu fdisk. Z knihy autorů Christiansen a
# Torkington, Perl Cookbook 5.


sub cut2fmt { my (@positions) = @_; my $template = ''; my $lastpos =
        1;
        foreach $place (@positions) {
                $template .= "A" . ($place - $lastpos) . " ";
                $lastpos = $place;

        }

    $template .= "A*"; return $template; }


# Sub gpl je subrutina (funkce), která dodává informace o licenci GPL a další # data záhlaví do aktuálního výstupního souboru.

sub gpl { my $FILE = shift; my $year = shift;

        print $FILE <<FINIS;

# Copyright $year až do data poslední změny Charles Curley.

# Další informace poskytne autor Charles Curley na # http://www.charlescurley.com/.

FINIS

}

sub getBootSector { my $infile = $_[0]; my $outfile = $_[1];

    $systemcmd = "dd if=$infile of=$outfile bs=512 count=1 &> /dev/null "; system ($systemcmd); }


# Pokud máte pouze jediný odkládací oddíl, musí to být on.
# Jinak si uživatel musí poradit sám. Skenuje se fstab na připojovací body
# odkládacích oddílů, které mají jmenovky pro svá zařízení. Pokud existuje
# pouze jediný, skript předpokládá, že je to ten správný, jinak prochází.


sub getswaplabel { my $dev = $_[0];

        $fstabpid = open (FSTAB, "< /etc/fstab")
                or die "Nelze zavolat fork: $!\n";
```

```
        while (defined (my $line = <FSTAB>)) {
                chop ($line);
                @fstabs = split (" ", $line);
                if (@fstabs[1] eq "swap") {

                        $swaplabel = @fstabs[0];
                        if ($swaplabel =~ /LABEL/) {
                        $swaps++;

                    $sl = substr ($swaplabel, 6); } # print ("\"@fstabs[0]\", \"@fstabs[1]\", \"$sl\", $swaps.\n");
                break;
                }
        }
        close (FSTAB);


# print "jmenovka je $sl.\n";

        if ($swaps == 1) {
                $ret = "mkswap \$blockcheck -L $sl";
                $ret .= " $dev\n\n";

            } else {
                  $ret = "mkswap \$blockcheck $dev\n\n";
                  }


# print ("Vráceno :$ret\n");

        return $ret; }

# dolvm je subrutina ke zpracování oddílů LVM. Tento kód # je experimentální...

$lvms = 0; # hodnota true při následujících průchodech

sub dolvm {

         print ("V dolvm ()...\n");

        if ($lvms == 0) {
                $lvms = 1;


                # V souboru /etc/fstab vyhledá logické svazky a zapíše skript # pro vytvoření systémů souborů na těchto svazcích a další skript # pro jejich
                pozdější připojení.

                $mklvs = open (MKLVS, "> make.lvs")
                        or die "Nelze zavolat fork: $!\n";


                print MKLVS <<FINIS; #! /bin/sh

# Skript k vytvoření systémů souborů na logických svazcích. Vytvořen při # obnovení od základů pomocí skriptu make.fdisk jazyka Perl. FINIS

                &gpl (*MKLVS, "2006");


                print MKLVS <<FINIS;

export blockcheck=\$1; if [ "\$blockcheck" != "-c" ] && [ -n "\$blockcheck" ]
then echo "\${0}: Vytvoří souborové systémy na logických svazcích." echo "\${0}: -c: kontrola bloků při vytváření souborových systémů." exit 1;
fi
```

```
export LVM_SYSTEM_DIR=\$(pwd)/lvm

FINIS

        $mtlvs = open (MTLVS, "> mount.lvs")
            or die "Nelze zavolat fork: $!\n";


        print MTLVS <<FINIS; #! /bin/sh

# Skript k připojení systémů souborů na logických svazcích. Vytvořen při # obnovení od základů pomocí skriptu make.fdisk jazyka Perl. FINIS

        &gpl (*MTLVS, "2006");


        # Nyní cyklicky prochází všechny známé logické svazky a nastaví # je. N.B.: Tento kód byl testován v počítači s jediným # logickým svazkem. *Měl by* však fungovat.

        $pvdisp = open (PVDISP, "pvdisplay -c |")
            or die ("Nelze otevřít zobrazení LVM.\n");

        while (defined (my $pv = <PVDISP>)) {
            chop ($pv);
            print ("$pv\n");
            @pv = split (":", $pv);
            $uid = @pv[11];
            $pvname = @pv[1];
            $phv = @pv[0];
            print ("pv $pvname má uid $uid.\n");


            # Zálohování podrobností lvm správce LVM. Načtení konf. souborů. system ("vgcfgbackup -f LVM.backs.$pvname $pvname");

            print (MKLVS "echo \"y\\n\" | pvcreate -ff --uuid \"$uid\"\\n"); print (MKLVS " --restorefile lvm/archive/${pvname}_*.vg $phv\n");
            print (MKLVS "vgcfgrestore --file LVM.backs.$pvname $pvname\n\n");
        }

            print (MKLVS "# Ošklivě závislé na distribuci!\nif [ -e /etc/init.d/lvm ] ; then\n");
    print (MKLVS " /etc/init.d/lvm start\nfi\n\n");

        $fstabpid = open (FSTAB, "< /etc/fstab")
            or die "Nelze zavolat fork: $!\n";

        while (defined (my $line = <FSTAB>)) {
            chop ($line);
            @fstabs = split (" ", $line);
            if (@fstabs[0] =~ /VolGroup/ ) {

                # print ("$line\n");
                if (@fstabs[2] eq "swap") { print (MKLVS "echo\necho změna LV @fstabs[0] na odkl. oddíl.\n"); print (MKLVS "mkswap \
                    $blockcheck @fstabs[0]\n\n");
                            } elsif (@fstabs[2] == "ext3") { print (MKLVS "echo\necho změna LV @fstabs[0],
                                @fstabs[1],"); print (MKLVS " na oddíl ext3.\n"); print (MKLVS "mke2fs -j \
                                $blockcheck @fstabs[0]\n\n");

                        print (MTLVS "mkdir -p /target$fstabs[1]\n"); print (MTLVS "mount @fstabs[0] /target
                        $fstabs[1]\n\n"); } elsif (@fstabs[2] == "ext2") { print (MKLVS "echo\necho změna LV
                        @fstabs[0], @fstabs[1],"); print (MKLVS " na oddíl ext2.\n"); print (MKLVS "mke2fs \
                        $blockcheck @fstabs[0]\n\n");

                    print (MTLVS "mkdir -p /target$fstabs[1]\n"); print (MTLVS "mount @fstabs[0] /target
                    $fstabs[1]\n\n"); } else { print ("Neznámý typ logického svazku @fstabs[0]\n"); }
            }
        }
```

```
            print (MTLVS "mount | grep -i \"/target\"\n");


            close (FSTAB);
            close (MKLVS);
            close (MTLVS);


            chmod 0700, "${outputfilepath}make.lvs";
            chmod 0700, "${outputfilepath}mount.lvs";


            # Kopírování konfigurace LVM na dostupné místo...

            system ("cp -rp /etc/lvm .");

        }

        print ("Ukončování dolvm ()...\n");

        return ($ret); }


# Začátek hlavního kódu.

# Poskytuje výchozí zařízení.

# print "\$ARGV[0] is $ARGV[0].\n";
$device = defined ($ARGV[0]) ? $ARGV[0] : "/dev/hda";

# Je nutné zkontrolovat, zda $device je symbolický odkaz. Pokud ano, je # připojovací bod cílem odkazu. (Mandrake) Jinak probíhá
hledání # připojovacích bodů na $device. Fedora, Red Hat.

if ( -l $device) {

        # Jedná se o symbolický odkaz. Skript načte cíl odkazu a pak jej změní na # absolutní cestu se zachováním číslování.

            $mountdev = '/dev/' . readlink ($device);
            $mountdev =~ s|ide/host(\d+)/bus(\d+)/target(\d+)/lun(\d+)/disc
            |ide/host\1/bus\2/target\3/lun\4|x;

} else { # Není to symbolický odkaz, bude pouze přiřazen. $mountdev = $device;
}

# print "Zařízení je $device; připojené zařízení je $mountdev.\n";

# Příprava formátovacího řetězce. Zde jsou podle mých zkušeností užitečné dva # formátovací řetězce. Čísla sloupců začínají od 1, tj. sloupec
nejvíce vlevo # je sloupec 1, nikoli 0 jako v programu Emacs.

# Výběr formátovacího řetězce závisí na verzi programu fdisk.

$fdpid = open (FDVER, "fdisk -v |") or die "Nelze zavolat fork: $!\n";
while (<FDVER>) { @_ = unpack ("A7 A*", $_); $fdver=$_[1]; $fdver =~ s/[^\d.]//g; # odfiltrování všeho mimo čísel a teček, jako v "2.12pre".
}


# print "Verze programu fdisk je $fdver\n";


if ($fdver < 2.12) {
# fdisk do verze 2.11?? Red Hat, Fedora Core 1

        $fmt = cut2fmt (11, 19, 24, 34, 45, 49); } else { # fdisk verze 2.12 a novější?? Mandrake 10.0, Fedora Core 2
        $fmt = cut2fmt (12, 14, 26, 38, 50, 55); } # print "Formátovací řetězec je $fmt.\n";
```

```
# definice prvků v poli @_.
$dev = 0;
$bootable = 1;
$firstcyl = 2;
$lastcyl = 3;
$parttype = 5;
$partstring = 6;


$target = "\/target";

$outputfilename = $device;
$outputfilename =~ s/\//./g;
$outputfilename = substr ($outputfilename, 1, 100);


$outputfilepath = "/root/bin/";



# Výpočet hodnoty hash jmenovek.
$mpid = open (MOUNT, "mount -l |") or die "Nelze zavolat fork: $!\n";
ßwhile (<MOUNT>) {

if ($_ =~ /^$mountdev/i) { # jedná se o řádek s oddílem?
#               print $_; # výpis pouze pro informaci split; if ($_[6] ne "") { # zpracuje, pouze pokud existuje jmenovka
            $_[6] =~ s/[\[\]]//g; # strike [ and ]. $labels{$_[0]} = $_[6]; # print "Jmenovka souborového zařízení $_[0] je
            $labels{$_[0]}.\n"; }


# Připojují se pouze systémy ext2fs nebo ext3fs pro čtení i zápis.

if ($_[4] =~ /ext[23]/ and $_[5] =~ /\(rw/ ) {

if ($_[0] =~ /ide/i) {

# V případě systému typu devfs, např. Mandrake, tento kód
# zajistí zpětný převod z označení devfs na starší
# označení /dev/hd* pro nástroj tomsrtb. NEVYZKOUŠEL jsem tento
# kód pro jiné jednotky než /dev/hda. Kód také
# nepracuje s jednotkami SCSI.

if ( $_[0] =~ /target0/ && $_[0] =~ /bus0/ ) {
$letter = 'a';
} elsif ( $_[0] =~ /target1/ && $_[0] =~ /bus0/) {
$letter = 'b';
} elsif ( $_[0] =~ /target0/ && $_[0] =~ /bus1/) {
$letter = 'c';
} else {
$letter = 'd';
}
$_[0] =~ s|/ide/host\d+/bus\d+/target\d+/lun\d+/part|/hd|g;
$_[0] =~ s/hd/hd$letter/;
}
$mountpoints{$_[2]} = $_[0];
#                       print "$_[2] je připojovací bod pro tomsrtbt";

#                       print " device $mountpoints{$_[2]}.\n";

                }
        }

}
```

close (MOUNT);

```
# Z□k v⌠stup programu sfdisk. Pokud je sfdisk dostupn⌠ při obnoven # (např. v distribuci Knoppix), použije se.
system "sfdisk -d $device > $outputfilepath${outputfilename}.sfd";

# Jinak se použije v⌠stup programu fdisk, kter⌠ může a nemusí
# b⌠t přesnějš.
$fpid = open (FDISK, "fdisk -l $device |") or die "Nelze zavolat fork: $!\n";
open (OUTPUT, "> $outputfilepath${outputfilename}")

    or die "Nelze otevř叩它 v⌠stupn soubor $outputfilepath${outputfilename}.\n"; while
(<FDISK>) {
if ($_ =~ /^$device/i) {                    # jedná se o řádek s oddílem?
#     print $_;                             # výpis pouze pro informaci

    chop;                                   # odstranění koncového \r

@_ = unpack ($fmt, $_);

# Nyn odfiltruje mezery z č□el cylindrů, mezery a
# poč疽ečn znaky plus z typu odd匀u.
@_[$firstcyl] =~ s/[ \t]+//;
@_[$lastcyl] =~ s/[ \t]+//;
@_[$parttype] =~ s/[+ \t]+//;

$partnumber = substr(@_[$dev], 8, 10); # načten č. odd匀u pro tento ř疆ek

        # pouze pro informaci # print " $partnumber, @_[$firstcyl], @_[$lastcyl],"; # print "
@_[$parttype], @_[$partstring]\n";

# Zde zač匡 generov疣 v⌠stupu pro obnoven odd匀u, # kter⌠ je uveden na tomto ř疆ku.

print OUTPUT "n\n";
if ($partnumber < 5) { # prim疵n odd匀 Linux if (@_[$parttype] == 83) {
print OUTPUT "p\n$partnumber\n@_[$firstcyl]\n"; # v př厓adě, že vše je na jednom cylindru if
(@_[$firstcyl] ne @_[$lastcyl]) {
    print OUTPUT "@_[$lastcyl]\n"; }

# Nyn zjist, zda se jedn o odd匀 se syst駑em ext3 # (s transakčn□ zpracov疣□ metadat). #
Princip je založen na v⌠pisu odd匀u pomoc dumpe2fs # a vyhled疣 řetězce "journal" programem
grep. Pokud je # odd匀 typu ext2, v⌠stup bude pr痙dn⌠. Jestliže se jedn o # ext3, nastav se
pomoc existuj癥加o v⌠stupu # parametr př勛azov駁o ř疆ku. Parametr př勛azov駁o ř疆ku # je
um□těn do asociativn加o pole (hash), takže nen # nutn jej resetovat na nulov⌠ řetězec # při
ukončen.
```

```
$dpid = open (DUMPE2FS, "dumpe2fs @_[$dev] 2>/dev/null | grep -i journal |")
                    or die "Nelze zavolat fork: $!\n"; while
                (<DUMPE2FS>) {
#                   print "Dumpe2fs: $_"; $ext3{$_[$dev]} =
                    "-j "; last;
                }
                close (DUMPE2FS);


                if ($labels{@_[$dev]}) { # je k dispozici jmenovka? $format .=
                    "echo\necho form疸ov疣 $checking@_[$dev]\n"; $format .= "mke2fs
                    $ext3{$_[$dev]}\$blockcheck"; $format .= " -L $labels{@_[$dev]}
                    @_[$dev]\n\n";
                } else { $format .= "echo\necho form疸ov疣 $checking@_[$dev]\n"; $format .=
                    "mke2fs $ext3{$_[$dev]}\$blockcheck @_[$dev]\n\n";
                }

                # rozšř冷n⌈ odd勻
            } elsif (@_[$parttype] == 5) { # print ("Prob朮 vytvořen rozšř冷n駇o odd勻u.\n");
                print OUTPUT "e\n$partnumber\n@_[$firstcyl]\n"; if (@_[$firstcyl] ne
                @_[$lastcyl]) {
                    print OUTPUT "@_[$lastcyl]\n";
                }


                # rozšř冷n⌈ odd勻, Win95 Ext (LBA)
            } elsif (@_[$parttype] eq "f") { # print ("Prob朮 vytvořen rozšř冷n駇o odd勻u
                LBA.\n"); print OUTPUT "e\n$partnumber\n@_[$firstcyl]\n"; if (@_[$firstcyl] ne
                @_[$lastcyl]) {
                    print OUTPUT "@_[$lastcyl]\n"; } $typechanges .= "t\n$partnumber\nf\n";

                # prim疵n odkl疆ac odd勻 Linux
            } elsif (@_[$parttype] == 82) { print OUTPUT "p\n$partnumber\n@_[$firstcyl]\n"; if
                (@_[$firstcyl] ne @_[$lastcyl]) {
                    print OUTPUT "@_[$lastcyl]\n"; } $typechanges .= "t\n$partnumber\n82\n";
                $format .= "echo\necho Změna @_[$dev] na odkl. odd勻.\n"; if
                ($labels{@_[$dev]}) { # je k dispozici jmenovka?
                  $format .= "mkswap \$blockcheck -L
                  $labels{@_[$dev]}"; $format .= "
                  @_[$dev]\n\n"; } else { $format .= getswaplabel
                  (@_[$dev]); }

                # Prim疵n odd勻 mess-dos. Skript nepracuje se skryt⌈mi # odd勻y.

            } elsif ( @_[$parttype] == 1 || @_[$parttype] == 4 || @_[$parttype] == 6 ||
                      @_[$parttype] eq "b" || @_[$parttype] eq "c"
              || @_[$parttype] eq "e" ) { # print ("Prob朮 vytvořen prim疵n朮o odd勻u DOS.\n");
```

```
        getBootSector (@_[$dev], "$outputfilepath$outputfilename$partnumber");

        print OUTPUT "p\n$partnumber\n@_[$firstcyl]\n";
        # v př厓adě, že vše je na jednom cylindru
        if (@_[$firstcyl] ne @_[$lastcyl]) {

            print OUTPUT "@_[$lastcyl]\n";
        }


        $typechanges .= "t\n$partnumber\n@_[$parttype]\n";
        $format .= "echo\necho form疸ov疣 $checking@_[$dev]\n";
        $format .= "mkdosfs \$blockcheck";
        if ( @_[$parttype] == b || @_[$parttype] == c) {

            # Je to odd匀 FAT32 syst駑u W9x. Přid疣 parametru př勛. ř疆ku.
            $format .= " -F 32";
        }
        $format .= " @_[$dev]\n";
        $format .= "# obnoven spouštěc刂o sektoru FAT.\n";
        $format .= "dd if=$outputfilename$partnumber";
        $format .= " of=@_[$dev] bs=512 count=1\n\n";


    } elsif ( @_[$parttype] == "8e") {
        $format .= dolvm ();

    } else {
        # jak⌈koli jin⌈ odd匀
        print OUTPUT "p\n@_[$firstcyl]\n";
        if (@_[$firstcyl] ne @_[$lastcyl]) {

            print OUTPUT "@_[$lastcyl]\n";
        }
        $typechanges .= "t\n$partnumber\n@_[$parttype]\n";

    }

} else { # logick⌈ odd匀 Linux if (@_[$parttype] == 83) {
          print OUTPUT "l\n@_[$firstcyl]\n";
          if (@_[$firstcyl] ne @_[$lastcyl]) {
          print OUTPUT "@_[$lastcyl]\n";
          }


        # Nyn zjist, zda se jedn o odd匀 se syst駑em ext3 # (s transakčn zpracov疣 metadat).
        # Princip je založen na v⌈pisu odd匀u pomoc dumpe2fs # a vyhled疣 řetězce "journal"
        programem grep. Pokud je # odd匀 typu ext2, v⌈stup bude pr痙dn⌈. Jestliže se jedn o #
```

```
ext3, nastav se pomoc existuj爥力乚o v⌠stupu # parametr př勛azov駁o ř疆ku. Parametr př
勛azov駁o ř疆ku # je um▯těn do asociativn力乚o pole (hash), takže nen # nutn jej resetovat
na nulov⌠ řetězec # při ukončen.

$dpid = open (DUMPE2FS,
                    "dumpe2fs @_[$dev] 2>/dev/null | grep -i journal |") or
          die "Nelze zavolat fork: $!\n"; while (<DUMPE2FS>) {
#           print "Dumpe2fs: $_"; $ext3{$_[$dev]}
            = "-j "; last;
        }
        close (DUMPE2FS);


        if ($labels{@_[$dev]}) { # je k dispozici jmenovka? $format .=
            "echo\necho form疸ov疣 $checking@_[$dev]\n"; $format .=
            "mke2fs $ext3{@_[$dev]}\$blockcheck"; $format .= " -L
            $labels{@_[$dev]} @_[$dev]\n\n";
        } else { $format .= "echo\necho form疸ov疣 $checking@_[$dev]\n"; $format .=
            "mke2fs $ext3{@_[$dev]}\$blockcheck @_[$dev]\n\n";
        }

        # logick⌠ odkl疆ac odd勻 Linux
    } elsif (@_[$parttype] == 82 ) {
        print OUTPUT "l\n@_[$firstcyl]\n";
        if (@_[$firstcyl] ne @_[$lastcyl]) {

            print OUTPUT "@_[$lastcyl]\n"; } $typechanges .= "t\n$partnumber\n82\n";
        $format .= "echo\necho Změna @_[$dev] na odkl. odd勻.\n"; if
        ($labels{@_[$dev]}) { # je k dispozici jmenovka?
          $format .= "mkswap \$blockcheck -L
          $labels{@_[$dev]}"; $format .= "
          @_[$dev]\n\n"; } else { $format .=
          getswaplabel (@_[$dev]); }

        # Logick⌠ odd勻 mess-dos. Skript nepracuje se skryt⌠mi # odd勻y.

    } elsif ( @_[$parttype] == 1 || @_[$parttype] == 4 || @_[$parttype] == 6 ||
              @_[$parttype] eq "b" || @_[$parttype] eq "c" || @_[$parttype] eq "e" ) {
# print ("Prob力乚 vytvořen logick駁o odd勻u DOS.\n");

        getBootSector (@_[$dev], "$outputfilepath$outputfilename$partnumber");

        print OUTPUT "l\n$partnumber\n@_[$firstcyl]\n"; # v př厓
        adě, že vše je na jednom cylindru if (@_[$firstcyl] ne
        @_[$lastcyl]) {
            print OUTPUT "@_[$lastcyl]\n"; } $typechanges .= "t\n
        $partnumber\n@_[$parttype]\n"; $format .= "echo\necho form疸ov疣
        $checking@_[$dev]\n"; $format .= "mkdosfs \$blockcheck"; if ( @_[$parttype]
```

```
                == b || @_[$parttype] == c) {
                    # Je to odd匀 FAT32 syst駑u W9x. Přid疣 parametru př勛. ř疆ku.
                    $format .= " -F 32";
}
$format .= " @_[$dev]\n";
$format .= "# obnoven spouštěc九o sektoru FAT.\n";
$format .= "dd if=$outputfilename$partnumber";
$format .= " of=@_[$dev] bs=512 count=1\n\n";


} elsif ( @_[$parttype] == "8e") { $format .= dolvm ();
} else { # jak⌈koli jin⌈ odd匀 print OUTPUT "l\n@_[$firstcyl]\n"; if (@_[$firstcyl] ne
    @_[$lastcyl]) {
    print OUTPUT "@_[$lastcyl]\n"; } $typechanges .= "t\n$partnumber\n@_[$parttype]\n";
    } }

 # zpracov疣 spouštěc燼h odd匀ů if (@_[$bootable] =~ /\*/) { print OUTPUT "a\n
 $partnumber\n"; }
} else { # Když se prov疆ěn skriptu dostalo až sem, aktu疝n ř疆ek # neobsahuje ž疆n⌈ odd匀. Je
    nutn z□kat geometrii pro fdisk. # Pak mus叱e vynutit, aby program fdisk použil aktu疝n #
    geometrii při obnoven. Tento ř疆ek označte na disku n疽troje # tomstrbt znakem komentř疇,
    protože jej n疽troj nepřij□.

if ($_ =~ /heads.*sectors.*cylinders/i) {
#               print $_;                   # opět pro informaci.
                chop;
                @geometry = split (/ /, $_);
                $geometry = "-H $geometry[0] -S $geometry[2] -C $geometry[4]";
#               print $geometry;
        }
    }
}

# Připoj všechny změny typů odd匀ů, ověř a vypš冷 # v⌈sledky.

print OUTPUT "${typechanges}v\nw\n";

close (OUTPUT); close (FDISK);


open (OUTPUT, "> ${outputfilepath}make.$outputfilename") or die "Nelze otevř叱 v⌈stupn
    soubor ${outputfilepath}make.$outputfilename.\n";

print OUTPUT <<FINIS; #! /bin/sh # Skript obnov data odd匀ů na pevn⌈ disk a form疽uje # tyto
odd匀y. Vytvořeno při obnoven od z疚ladů pomoc skriptu # make.fdisk jazyka Perl. FINIS

&gpl (*OUTPUT, "2001");

print OUTPUT <<FINIS;
```

```
swapoff -a # Ošklivě z疱isl na distribuci! if [ -e /etc/init.d/lvm ] ; then
    /etc/init.d/lvm stop fi

export blockcheck=\$1;

if [ "\$blockcheck" != "-c" ] && [ -n "\$blockcheck" ]
then echo "\${0}: automatizovan obnoven bez interakce s uživatelem." echo "\${0}: -c: kontrola
    bloků při vytvř疇n souborov⌈ch syst駑ů." exit 1;
fi


FINIS


# Vyčist starou tabulku odd匀ů. Vypne odkl疆ac odd匀 v př厓adě, že se
# použ咩.


print OUTPUT "dd if=/dev/zero of=$device bs=512 count=2\n\nsync\n\n";



# př勛az pro fdisk


$fdiskcmd .= "# kontrola, zda je k dispozici sfdisk. Pokud ano, použije se.\n";$fdiskcmd .= "if
which sfdisk ; then\n";$fdiskcmd .= " echo \"Použ咩 se sfdisk.\"\n";$fdiskcmd .= " sfdisk --force
$geometry $device < ${outputfilename}.sfd\n";$fdiskcmd .= "else\n";$fdiskcmd .= " echo \"Použ
咩 se fdisk.\"\n";$fdiskcmd .= " fdisk $geometry $device \< $outputfilename\n";$fdiskcmd .=
"fi\n\nsync\n\n";


print OUTPUT $fdiskcmd;
print OUTPUT $format;


print OUTPUT "fdisk -l \"$device\"\n";


close (OUTPUT);


# Nyn generuje skript, kter⌈ vytvoř připojovac body na kořenov駑
# odd匀u a jin⌈ch odd匀ech.

open (OUTPUT, "> ${outputfilepath}mount.$outputfilename") or die "Nelze otevř咱 v⌈stupn
    soubor ${outputfilepath}make.$outputfilename.\n";

print OUTPUT <<FINIS; #! /bin/sh
```

Skript vytvoř minim疝n adresř痂v⌈ strom na c匀ov駑 pevn駑 disku # a připoj k němu odd匀y. Vytvořeno při obnoven od z疚ladů pomoc # skriptu make.fdisk jazyka Perl. FINIS

&gpl (*OUTPUT, "2001");

print OUTPUT <<FINIS;

VAROVヂN^: Pokud syst駑 Linux připojuje odd匀y přes hranice # pevn駁o disku, potřebujete v燼e skriptů "mount.dev.*. Je nutn zajistit, # aby byly spuštěny ve spr疱n駑 pořad. Kořenov⌈ odd匀 je potřeba # připojit jako prvn a pak zb⌈vaj燼 odd匀y v pořad jejich větven. Jestliže # se připojuj křž邵vě, mus吪e to prov駸t ručně.

FINIS

K dispozici je hodnota hash připojovac燼h bodů a zař坦en v %mountpoints. Je # však nutn je zpracovat, aby adresř疇 vznikly na # spr疱n駑 c匀ov駑 odd匀u. Je-li např. adresř /usr/local na sv駑 vlastn⊟# odd匀u, je nutn nejdř咩e připojit /usr a pak vytvořit /usr/local. Lze to # zajistit seřazen⊟seznamu. Kratš cesty k připojovac▽ bodům budou vytvořeny # jako prvn. Hodnoty hash nelze řadit př⊟ɵ, a použije se proto pole.

Skript generuje př勛azy pro vytvořen př□lušn⌈ch připojovac燼h bodů a pak # připoj odd匀y k připojovac⊟bodům. Jedn se o př厓avu na rozbalen # obsahu disku ZIP programem tar, kter zajišuje skript restore.metadata.

```
foreach $point ( sort keys %mountpoints) { print OUTPUT "\n# $point je připojovac bod pro";
    print OUTPUT " tomsrtbt device $mountpoints{$point}.\n"; print OUTPUT "mkdir -p $target
    $point\n"; print OUTPUT "mount $mountpoints{$point} $target$point\n";
}

print OUTPUT "\nmount | grep -i \"/target\"\n";

close (OUTPUT);
```

Tyto skripty jsou nebezpečn a měly by b⌈t viditeln jen uživateli root.

```
chmod 0700, "${outputfilepath}make.$outputfilename"; chmod
0700, "${outputfilepath}mount.$outputfilename"; chmod 0600, "$
{outputfilepath}${outputfilename}*";
```

make.dev.hda

Tento skript je ukázkou výstupu výše uvedeného skriptu make.fdisk. Používá datové soubory jako dev.hda. Vytvoří oddíly a na některé z nich umístí souborové systémy. Jedná se o první skript spuštěný během obnovení.

Pokud si troufáte upravit skript dev.hda, např. kvůli přidání nového oddílu, může být nutné přizpůsobit i tento skript.Chcete-li, aby skript make.dev.hda při umístění souborových systémů na oddíly kontroloval chyb-né bloky, použijte parametr příkazového řádku -c.

```
#! /bin/sh

# Skript obnoví data oddílů na pevný disk a formátuje # tyto oddíly. Vytvořeno při obnovení od základů pomocí skriptu #
make.fdisk jazyka Perl.

# Copyright 2001 až do data poslední změny Charles Curley.
```

```
# Další informace poskytne autor Charles Curley na # http://www.charlescurley.com/.


export blockcheck=$1;

if [ "$blockcheck" != "-c" ] && [ -n "$blockcheck" ]
then echo "${0}: automatické obnovení bez interakce s uživatelem." echo "${0}: -c: kontrola bloků při vytváření souborových systémů." exit 1;
fi

dd if=/dev/zero of=/dev/hda bs=512 count=2

swapoff -a sync

# kontrola, zda je k dispozici sfdisk. Pokud ano, použije se.
if which sfdisk ; then echo "Používá se sfdisk." sfdisk -H 128 -S 63 -C 523 /dev/hda < dev.hda.sfd
else echo "Používá se fdisk." fdisk -H 128 -S 63 -C 523 /dev/hda < dev.hda
fi

sync

echo echo Formátování /dev/hda1 mkdosfs $blockcheck /dev/hda1 # obnovení spouštěcího sektoru FAT. dd if=dev.hda1 of=/dev/hda1 bs=512 count=1
echo echo Formátování /dev/hda2 mke2fs -j $blockcheck -L /boot /dev/hda2

echo echo Formátování /dev/hda3 mke2fs -j $blockcheck -L / /dev/hda3

echo Změna /dev/hda5 na odkládací oddíl. mkswap $blockcheck /dev/hda5

fdisk -l "/dev/hda"
```

make.lvs

Skript make.lvs je generován skriptem make.fdisk, ale pouze v případě, že jsou dostupné logic-ké svazky. Jak vyplývá z názvu, vytvoří logické svazky a umístí na ně souborové systémy.

```
#! /bin/sh

# Skript k vytvoření systémů souborů na logických svazcích. Vytvořen při # obnovení od základů pomocí skriptu make.fdisk jazyka Perl.

# Copyright 2006 až do data poslední změny Charles Curley.

# Další informace poskytne autor Charles Curley na # http://www.charlescurley.com/.


export blockcheck=$1;

if [ "$blockcheck" != "-c" ] && [ -n "$blockcheck" ]
then echo "${0}: Vytvoří souborové systémy na logických svazcích." echo "${0}: -c: kontrola bloků při vytváření souborových systémů." exit 1;
fi

export LVM_SYSTEM_DIR=$(pwd)/lvm.cfg

    echo "y\n" | pvcreate -ff --uuid "CCmw0N-0We2-HzRS-jRZa-FkC7-NxTc-oAfvpX"\ --restorefile
            lvm.cfg/archive/VolGroup00_*.vg /dev/hda3 vgcfgrestore --file LVM.backs VolGroup00

  # Ošklivě závislé na distribuci! if [ -e /etc/init.d/lvm ] ;
  then /etc/init.d/lvm start fi

echo echo Změna log. svazku /dev/VolGroup00/LogVol00 na oddíl ext3. mke2fs -j $blockcheck /dev/VolGroup00/LogVol00

echo echo Změna log. svazku /dev/VolGroup00/LogVol02 na oddíl ext3. mke2fs -j $blockcheck /dev/VolGroup00/LogVol02

echo echo Změna log. svazku /dev/VolGroup00/LogVol01 na odkládací oddíl. mkswap $blockcheck /dev/VolGroup00/LogVol01
```

mount.dev.hda

Tento skript je ukázkou výstupu výše uvedeného skriptu make.fdisk. Vytvoří připojovací body a připojí k nim oddíly, aby byl cílový souborový systém připraven na obnovení souborů. Jedná se o druhý skript spuštěný během obnovení.

Pokud si troufáte upravit skript dev.hda, např. kvůli přidání nového oddílu, může být nutné při-způsobit i tento skript.

```
#! /bin/sh


# Skript vytvoří minimální adresářový strom na cílovém pevném disku
# a připojí k němu oddíly. Vytvořeno při obnovení od základů pomocí
# skriptu make.fdisk jazyka Perl.


# Copyright 2001 až do data poslední změny Charles Curley.


# Další informace poskytne autor Charles Curley na
# http://www.charlescurley.com/.



# VAROVÁNÍ: Pokud systém Linux připojuje oddíly přes hranice
# pevného disku, potřebujete více skriptů "mount.dev.*. Je nutné zajistit,
# aby byly spuštěny ve správném pořadí. Kořenový oddíl je potřeba
# připojit jako první a pak zbývající oddíly v pořadí jejich větvení. Jestliže
# se připojují křížově, musíte to provést ručně.



# / je připojovací bod pro zařízení /dev/hda3 nástroje tomsrtbt.
mkdir /target/
mount /dev/hda3 /target/


# /boot je připojovací bod pro zařízení /dev/hda2 nástroje tomsrtbt.
mkdir /target/boot
mount /dev/hda2 /target/boot


mount | grep -i "/dev/hda"
```

mount.lvs

Skript mount.lvs je generován skriptem make.fdisk, ale pouze v případě, že jsou dostupné logické svazky. Jak vyplývá z názvu, připojuje logické svazky připravené na obnovení.

```
#! /bin/sh

# Skript k připojení systémů souborů na logických svazcích. Vytvořen při # obnovení od základů pomocí skriptu make.fdisk jazyka Perl.

# Copyright 2006 až do data poslední změny Charles Curley.
# Další informace poskytne autor Charles Curley na # http://www.charlescurley.com/.

mkdir -p /target/ mount /dev/VolGroup00/LogVol00 /target/

mkdir -p /target/home mount /dev/VolGroup00/LogVol02 /target/home

mount | grep -i "/target"
```

dev.hda

Tento datový soubor se používá při obnovení. Slouží jako vstup programu fdisk, kterému jej poskytuje skript make.dev.hda. Vytváří jej při zálohování skript make.fdisk. Pokud umíte pra-covat s programem fdisk, můžete si všimnout, že každý řádek obsahuje příkaz nebo hodnotu pro-gramu fdisk, např. číslo cylindru. Proto je možné úpravou tohoto souboru změnit velikosti

oddí-lů a přidat nové oddíly. Předposledním příkazem je tedy v, který ověří tabulku oddílů před tím, než je zapsána.

n p 1 1 29 a 1 n p 2 30 44 n e 3
45 1023 n l 45 944 n l 945 1023 t
1 6 t 6 82 v w

save.metadata

Jedná se o první skript spuštěný v rámci procesu zálohování. Volá výše uvedený skript make.fdisk. Pokud potřebujete zálohovat pevný disk SCSI nebo více pevných disků, upravte pří-slušným způsobem volání skriptu make.fdisk.

```
#! /bin/sh

# Skript pro uložení určitých metadat ze spouštěcího oddílu. Užitečný při # obnovení.

# Časové razítko: <2006-04-05 20:37:09 ccurley save.metadata>

# Copyright 2000 až do data poslední změny Charles Curley.

# Další informace poskytne autor Charles Curley na # http://www.charlescurley.com/.


# Crunch: Funkce pro kompresi obsahu adresáře a umístění # archivu na disk ZIP.

# Prvním parametrem je název archivního souboru,
# který chcete vytvořit. Před název se doplní zálohovací umístění $zip a
# za název se uvede přípona "tar.bz2".


# Všechny následující parametry budou považovány za další adresáře # nebo soubory, které chcete umístit do archivu.

function crunch {

if [ -z "$1" ] || [ -z "$2" ] # Kontrola, zda má parametr #1 či #2 nulovou délku. then
echo "-Parameter #1 nebo #2 chybí.-" # Také zda není předán žádný parametr.
return 1
else
local file=$1                   # Archivní soubor pro vytvoření
shift                           # Zahození názvu souboru
local dirs=$@                   # Adresář či adresáře k archivaci

local tarcmd="tar --numeric-owner -cjf" # Př勛az tar.


local tarit="$tarcmd $zip/$file.tar.bz2 $dirs"echo $tarit$tarit # provede operaci!

error=$? # Uložen k❿u ukončení


if [ $error != 0 ] # Došlo k chybě?

then # Ano

echo "Př勛az tar skončil ne俍pěšně s chybou $error"
echo $tarcmd $zip/$file.tar.bz2 $dirs
    exit $error # vr疸 k❿ ukončen programu tar fi
return 0                                # Kvůli případnému testování chyb.
fi
}
```

```
# Začátek hlavního kódu
export zip="/var/bare.metal";                   # Umístění archivů. Sem nepatří jednotka ZIP.
#     export save="/mnt/save";

RPMVABACKS=/etc                                 # kde jsou uloženy zálohy
RPMVAROOT=rpmVa                                 # název záloh s balíčky

ANC=${RPMVABACKS}/${RPMVAROOT}.anc # n痙ev nejstarš z疝ohyOLD=$
{RPMVABACKS}/${RPMVAROOT}.old # n痙ev středně star z疝ohyNEW=$
{RPMVABACKS}/${RPMVAROOT}.txt # n痙ev nejnovějš z疝ohy

if [ -f ${ANC} ]; then
echo "Odstraňov疣 ${ANC}"
rm ${ANC}
fi


if [ -f ${OLD} ]; then
echo "Snž冷n aktu疝nosti ${OLD}"
mv ${OLD} ${ANC}
fi


if [ -f ${NEW} ]; then
echo "Snž冷n aktu疝nosti ${NEW}"
mv ${NEW} ${OLD}
fi



# Nyn skript ulož informace o pevn駑 disku. Na každ駑 disku spuste skript
# make.fdisk v pořad připojen ke kořenov駑u odd勻u. Tj.
# spuste skript nejdř咩e na pevn駑 disku s kořenov⌠m odd勻em, pak
# na libovoln⌠ch disc癥h, kter se připojuj k prvn□u pevn駑u disku, d疝e
# na disc癥h, kter se připojuj k nim. Pokud m疸e např. kořenov⌠ odd勻
# na /dev/sdc, spuste nejdř咩e "make.fdisk /dev/sdc".


# Skript make.fdisk totiž generuje skript pro vytvoření
# připojovac癥h bodů a pak k nim připoj př□lušn odd勻y během
# prvn f痙e obnoven. Připojovac body je nutn vytvořit na odd勻u,
# kde budou um□těny. Odd勻y je nutn připojit v tomto
# pořad. M疸e-li např. adresř疇 /var a /var/ftp na odlišn⌠ch
# odd勻ech, mus咤e připojit /, vytvořit /var, pak připojit /var
# a nakonec vytvořit /var/ftp. Pořad, ve kter駑 skript "first.stage"
# spoušt připojovac skripty, z疱is na času jejich vytvořen.


# V př厓adě potřeby vložte mezi vol疣 skriptu make.fdisk ř疆ek "sleep 1".
```

```
echo "Uložen informac o pevn駭 disku"
make.fdisk /dev/hda


# z疝ohov疣 metadat RPM


echo "Ověřov疣 balč勛ů RPM."


rpm -Va | sort -t ' ' -k 3 | uniq > ${NEW}
echo "Ověřov疣 RPM bylo dokončeno, nyn se připojuje jednotka ZIP."
# Zkontroluje, zda je připojena jednotka ZIP.


# umount $zip
# modprobe ppa                  # Ovladač 100MB disku ZIP pro paralelní port
# mount $zip                    # Měl by mít ext2fs na oddílu 1.

# kompletní vyčištění
# rm -r $zip/*

# mkdir -p $zip/lost+found

# Protože se neukl疆 na disk ZIP, sn啫 se aktu疝nost m□tn kopie. rm -r $zip.old mv $zip $zip.old mkdir $zip

echo -e "$(hostname) disk ZIP pro obnoven od z疚ladů, vytvořen $(date)" > $zip/README.txt
uname -a >> $zip/README.txt

# Uchov informace o verzi. Testov疣o s distribuc Red Hat/Fedora, mělo by # fungovat v SuSE, Mandrake a jin⌈ch syst駭ech založen⌈ch na RPM. Vytvoř někdo # ekvivalent pro Debian?

for releasefile in $(ls /etc/*release*) ; do # echo $releasefile if [ -e $releasefile ] && [ ! -L
  $releasefile ] ; then
  cat $releasefile >> $zip/README.txt fi done

echo "Vytvořen z疝oh na jednotku ZIP."

# Slouž pro př厓ad, že bude nutn se na ně odkazovat při obnově. Proces # obnovy by měl b⌈t převž疣ě automatizov疣, ale jistota # je jistota...

fdisk -l /dev/hda > $zip/fdisk.hda

ls -al /mnt > $zip/ls.mnt.txt ls -al / > $zip/ls.root.txt

cd /

# Vytvoř minim疝n archivy na disku ZIP. Nejspš jsou # požadov疣y pro pozdějš obnoven.
```

```
crunch root --exclude root/.cpan --exclude root/.mozilla --exclude root/down root crunch boot
boot crunch etc --exclude etc/samba --exclude etc/X11 --exclude etc/gconf etc crunch lib lib
crunch usr.sbin usr/sbin
crunch usr.bin --exclude usr/bin/emacs-x --exclude usr/bin/emacs-21.4-x\ --exclude
   usr/bin/emacsclient --exclude usr/bin/emacs-nox --exclude\ usr/bin/gs --exclude usr/bin/pine --
   exclude usr/bin/gimp-1.2\ --exclude usr/bin/doxygen --exclude usr/bin/postgres --exclude\
   usr/bin/gdb --exclude usr/bin/kmail --exclude usr/bin/splint\ --exclude usr/bin/odbctest --
   exclude usr/bin/php --exclude \ usr/bin/xchat --exclude usr/bin/gnucash --exclude
   usr/bin/pdfetex\ --exclude usr/bin/pdftex --exclude usr/bin/smbcacls\ --exclude
   usr/bin/evolution-calendar --exclude usr/bin/xpdf\ --exclude usr/bin/xmms usr/bin
crunch sbin sbin
crunch bin bin
crunch dev dev


# RH8. Fedora 1 je umisuje do /lib
# crunch kerberos usr/kerberos/lib/


# N疽leduj灒 volitelně ukl疆an data.


# určeno pro program arkeia:
# crunch arkeia usr/knox


# uložen těchto dat, aby bylo možn obnovovat pomoc ssh. *crack*
# pro ověřov疣 při přihlš疇n k RH 7.0.
# RH 8.0
# crunch usr.lib usr/lib/*crack* usr/lib/libz* usr/lib/libssl* usr/lib/libcrypto*
# Fedora 1
# crunch usr.lib usr/lib/*crack* usr/lib/libz* usr/lib/libwrap*\
# usr/lib/libk* usr/lib/*krb5* /usr/lib/libgss*
# Fedora 3
crunch usr.lib usr/lib/*crack* usr/lib/libz* usr/lib/libwrap*\

  usr/lib/libk* usr/lib/*krb5* usr/lib/libgss*

# Grub vyžaduje tyto hodnoty při instalaci.
crunch usr.share.grub usr/share/grub


# uložen skriptů, pomoc nichž byl vytvořen disk ZIP, a skriptů, které
# umožn jeho obnoven.
mkdir $zip/root.bin
cp -p /root/bin/* $zip/root.bin
rm $zip/root.bin/*~ $zip/root.bin/#*#
```

echo "Testov疣 v⌠sledků."
find $zip -iname "*.bz2" | xargs bunzip2 -t

Nejde o norm疝n souč疽t procesu: disk ZIP je duplikov疣 na
připojen⌠ syst駑 NFS v jin駑 um□těn.

echo "Z疝ohov疣 jednotky ZIP na připojen⌠ syst駑 NFS."

umount $save# mount $save
rm -r $save/zip # mkdir -p $save/zip # cp -pr $zip $save

Protože skript pracuje se syst駑ov⌠mi daty, vytvoř ISO obraz spustiteln馼o # disku vhodn⌠ k vyp疝en. Použ咩 aktu疝n j疆ro.

mkbootdisk --iso --device $zip/bootdisk.$(uname -r).iso $(uname -r)

du -hs ${zip}* df -m

restore.metadata
Tento skript obnoví metadata z disku ZIP v rámci první fáze obnovení.

```
#! /bin/sh

# Skript pro obnovení metadat z disku ZIP. Lze jej spustit v rámci # nástroje tomsrtbt pouze po obnovení oddílů, vytvoření # a připojení systémů souborů. Předpokládá také, že disk ZIP byl již # připojen. Nejspíš je vhodné připojit disk ZIP pouze pro čtení.

# Časové razítko: <2006-04-05 20:36:49 ccurley restore.metadata>

# Copyright 2000 až do data poslední změny Charles Curley.

# Další informace poskytne autor Charles Curley na # http://www.charlescurley.com/.

umask 0000

cd ..                   # Předpokládá se umístění root.bin
zip=$(pwd);             # Místo připojení jednotky ZIP.
target="/target";       # Místo připojení pevného disku k obnovení.

ls -lt $zip/*.bz2       # Užitečné informace pro uživatele.

cd $target
```

# Obnoven archivn鏸h souborů metadat.
for archive in $(ls $zip/*.bz2); do
echo $archive
ls -al $archive
  bzip2 -dc $archive | tar -xf done

Vytvořen připojovac鏸h bodů pro druhou f痙i obnoven a dalš # č鋹ly.

Pokud spoušt咤e pomoc initrd, nezapomeňte zde vytvořit adresř, aby # mohlo j疆ro připojit initrd při spuštěn. tmp/.font-unix slouž pro # font-server xfs.

```
for dir in mnt/dosc mnt/zip mnt/imports mnt/nfs proc initrd tmp/.font-
 unix\ var/empty/sshd var/log back selinux sys /var/cache/yum /var/lock;
 do mkdir -p $target/$dir done

for dir in mnt usr usr/share $(ls -d var/*) selinux usr/lib var var/cache/yum; do chmod go-w
  $target/$dir
done

chown root:lock /var/lock chmod
775 /var/lock

# [root@jhereg /]# ll -d mnt usr usr/share $(ls -d var/*) selinux usr/lib var
var/cache/yum # drwxr-xr-x 4 root root # drwxr-xr-x 2 root root # drwxr-xr-x 14 root root #
drwxr-xr-x 40 root root # drwxr-xr-x 63 root root # drwxr-xr-x 20 root root # drwxr-xr-x 2 root
root # drwxr-xr-x 4 root root # drwxr-xr-x 4 root root # drwxr-xr-x 3 netdump netdump # drwxr-
xr-x 3 root root # drwxr-xr-x 3 root root # drwxr-xr-x 13 root root # drwxr-xr-x 2 root root #
drwxrwxr-x 4 root lock # drwxr-xr-x 7 root root # lrwxrwxrwx 1 root root # drwxr-x---4 root
named # drwxr-xr-x 2 root root # drwxr-xr-x 2 root root # drwxr-xr-x 2 root root # drwxr-xr-x 2
root root # drwxr-xr-x 13 root root # drwxr-xr-x 13 root root # drwxrwxrwt 2 root root # drwxr-
xr-x 3 root root
 4096 Oct 10 08:55 mnt
 4096 Oct 10 08:41 selinux
 4096 Oct 10 08:46 usr

12288 Oct 10 10:40 usr/lib 4096 Oct 10 11:11 usr/share 4096 Oct 10 08:52 var 4096 Oct 10
 08:51 var/account 4096 Oct 10 08:53 var/cache 4096 Oct 10 10:44 var/cache/yum 4096 Aug 22
 13:13 var/crash 4096 Oct 10 08:51 var/db 4096 Oct 10 08:52 var/empty 4096 Oct 10 11:11
 var/lib 4096 May 22 22:28 var/local 4096 Sep 1 08:37 var/lock 4096 Oct 10 11:14 var/log
    10 Oct 10 08:42 var/mail -> spool/mail 4096 Aug 22 14:33 var/named 4096 May 22 22:28
 var/nis 4096 May 22 22:28 var/opt 4096 May 22 22:28 var/preserve 4096 Mar 28 2005
 var/racoon 4096 Oct 10 11:14 var/run 4096 Oct 10 08:53 var/spool 4096 Oct 10 11:14 var/tmp
 4096 Oct 10 08:53 var/yp

# chmod a-w $target/proc # Obnoven opr疱něn pouze pro čten adresř疇 /proc


# Nastaven režimů
chmod 0111 $target/var/empty/sshd


# Pro distribuci Fedora. Prvn dva ř疆ky pro xfs.
# chroot $target chown xfs:xfs /tmp/.font-unix
# chmod 1777 $target/tmp/.font-unix # Nastaven sticky bitu.
chmod 1777 $target/tmp


# Obnoven skriptů, pomoc nichž byl vytvořen disk ZIP, a skriptů, které
# umožn jeho obnoven. Mělo by se jednat o nejdokonalejš skripty pro př厓ad,
```

že bude nutn během prvn f座e obnoven něco upravovat.

cp -p $zip/root.bin/* $target/root/bin

Nyn nainstalujte spouštěc sektor. # chroot
$target /sbin/lilo -C /etc/lilo.conf chroot $target /sbin/grub-
install /dev/hda

df -m

first.stage
Tento skript spustí celou první fázi obnovení bez zásahu operátora.Chcete-li, aby skript při umístění souborových systémů na oddíly kontroloval chybné bloky, pou-žijte parametr příkazového řádku -c.

```
#! /bin/sh


# Hlavní skript spouštějící jiné specializované skripty. Použijte tento skript
# pouze v případě, že chcete provést obnovení zcela automaticky. Jediným
# parametrem je -c. Vynutí kontrolu chybných bloků při formátování
# oddílů.


# Časové razítko: <2006-04-05 20:35:39 ccurley first.stage>


# Copyright 2002 až do data poslední změny Charles Curley.


# Další informace poskytne autor Charles Curley na
# http://www.charlescurley.com/.


# 2005-08-07 Nadále se již nepředpokládá pracovní adresář. Důvodem je,
# že pracovní adresář se může značně lišit podle
# použité distribuce a způsobu obnovení.


export blockcheck=$1;


if [ "$blockcheck" != "-c" ] && [ -n "$blockcheck" ]

then echo "${0}: automatické obnovení bez interakce s uživatelem." echo "${0}: -c: kontrola bloků při vytváření souborových systémů." exit 1;
fi

for drive in $( ls make.dev.* ); do echo $drive$'\a' sleep 2 ./$drive $blockcheck;
done

# Pokud existují nějaké svazky LVM, je nyní čas na jejich obnovení.

if [ -e LVM.backs ] && [ -e make.lvs ] && [ -e mount.lvs ]
then echo make.lvs$'\a' sleep 2 ./make.lvs
    echo mount.lvs$'\a' ./mount.lvs fi


# VAROVÁNÍ: Pokud systém Linux připojuje oddíly přes hranice # pevného disku, potřebujete více skriptů "mount.dev.*. Je nutné zajistit, # aby byly spuštěny ve správném pořadí, což následující smyčka nemusí # splnit. Kořenový oddíl je nutné připojit jako první a pak zbývající oddíly # v pořadí jejich větvení. Jestliže se připojují křížově, musíte to provést # ručně. Pracujete-li se správci LVM, je nutné postupovat # úplně jinak.

# Příkaz "ls -tr" vypíše skripty v pořadí jejich vytvoření, takže # je nejspíš vhodné vytvořit je (ve skriptu save.metadata) # v
```

pořadí, v jakém je chcete spouštět.

```
for drive in $( ls -tr mount.dev.* ); do echo $drive$'\a' sleep 2 ./$drive;
done

./restore.metadata

# Pokud si opravdu věříte, můžete odstranit následující symbol komentáře. # reboot
```

Druhá fáze

Tyto skripty jsou spuštěny v počítači při jeho zálohování nebo obnovení.

back.up.all

Tento skript ukládá na jiný počítač pomocí připojení NFS. Můžete jej přizpůsobit tak, aby ukládal na páskové jednotky nebo jiná média.

```
#! /bin/sh

# Zálohuje celý systém na disk jiného počítače. Kvůli této funkci # je nutné, aby měl vzdálený počítač dostatek volného místa na
disku, # které lze připojit přes nfs jako /mnt/save.

# Časové razítko: <2003-04-24 09:56:05 ccurley back.up.all>

# Copyright 2000 až do data poslední změny Charles Curley.

# Další informace poskytne autor Charles Curley na # http://www.charlescurley.com/.

save="/mnt/save"

# Kontrola přítomnosti umount $save mount $save

cd /

rm $save/tester.tar.old.gz mv $save/tester.tar.gz $save/tester.tar.old.gz

# uložení všeho kromě /mnt, /proc a adresářů připojených přes nfs.

time tar cf - / --exclude /mnt --exclude /proc --exclude $save\ | gzip -c > $save/tester.tar.gz
```

back.up.all.ssh

Skript má přesně stejnou funkci jako back.up.all, ale místo nfs používá ssh.

```
#! /bin/sh


# Zálohuje celý systém na disk jiného počítače. Kvůli této funkci
# je nutné, aby měl vzdálený počítač dostatek volného
# místa na disku. Tato verze při přenosu používá ssh a komprimuje
# pomocí bz2. To znamená, že tento skript potřebuje více informací
# o druhém počítači, což komplikuje modularizaci.


# Časové razítko: <2003-04-24 09:56:52 ccurley back.up.all.ssh>


# Copyright 2000 až do data poslední změny Charles Curley.


# Další informace poskytne autor Charles Curley na
# http://www.charlescurley.com/.


save="/backs/tester"
backup_server="charlesc"
```

```
# rotace starých záloh. Všechny operace jsou na jediném řádku, aby se minimalizovala
režie s ověřováním.
ssh $backup_server "rm $save/tester.tar.old.bz2; mv $save/tester.tar.bz2 \

        $save/tester.tar.old.bz2"

# uložení všeho kromě /mnt, /proc a adresářů squid.

time tar cf - / --exclude /mnt --exclude /proc --exclude /var/spool/squid\ | ssh $backup_server "bzip2 -9 > $save/tester.tar.bz2"
```

restore.all

Jedná se o skript pro obnovení, který se používá v případě zálohování pomocí skriptu back.up.all.

```
#! /bin/sh

# Skript obnoví všechna data z připojené položky nfs. Jedná se o poslední # fázi obnovení.

# Časové razítko: <2003-04-24 09:58:51 ccurley restore.all>
# Copyright 2000 až do data poslední změny Charles Curley.


# Další informace poskytne autor Charles Curley na
# http://www.charlescurley.com/.


export save="/mnt/save"


mount $save


cd /
gunzip -dc $save/tester.tar.gz | tar -xpkf


rm /var/run/*.pid


lilo
```

restore.all.ssh

Jedná se o skript pro obnovení, který se používá v případě zálohování pomocí skriptu back.up.all.ssh.

```
#! /bin/sh

# Skript obnoví všechna data pomocí ssh a bunzip2. Jedná se o # poslední fázi obnovení.

# Copyright 2000 až do data poslední změny Charles Curley.

# Časové razítko: <2003-04-24 09:59:10 ccurley restore.all.ssh>

# Další informace poskytne autor Charles Curley na # http://www.charlescurley.com/.

save="/backs/tester/" backup_server="charlesc"

cd /

ssh $backup_server "cat $save/tester.tar.bz2" | bunzip2 | tar -xpkf

rm /var/run/*.pid

lilo
```

Skripty zálohovacího serveru

Výše uvedené skripty ssh přinášejí možný bezpečnostní problém. Pokud je spustíte na firewallu, musí mít firewall přístup přes ssh k zálohovacímu serveru. V tomto případě může chytrý cracker proniknout i na zálohovací server. Bylo by bezpečnější spustit skripty pro zálohování a obnovení na zálohovacím serveru a zpřístupnit zálohovacímu serveru firewall. K tomu slouží tyto skripty. Přejmenujte je na get.x a restore.x, kde x je název cílového počítače. Upravte je (inicializaci proměnné $target), aby pracovaly s názvem hostitele cílového počítače, nebo je přepište tak, aby přijímaly argument příkazového řádku.
Tyto skripty zálohují a obnovují cílový počítač kompletně – neomezují se pouze na první fázi zálo-hování a obnovení. Všimněte si také, že skript get.tester také zálohuje disk ZIP pro případ, že potřebujete nahradit vadný disk ZIP.

Tyto skripty rutinně používám.

et.tester

```
#! /bin/sh


# Zálohování jednotky jiného počítače do tohoto systému. Kvůli této funkci
# je nutné, aby měl tento počítač dostatek volného místa na disku. Tato
# verze přenáší data přes ssh a komprimuje pomocí bz2. Tato
# verze byla vyvinuta proto, aby se zálohovaný systém neověřoval
# pro přihlášení k zálohovacímu počítači. Tento skript
# je určen k použití na firewallu. Není vhodné, aby se firewall
# ověřoval na zálohovacím serveru pro případ, že je firewall napaden útočníkem.


# Časové razítko: <2006-04-05 20:36:00 ccurley get.tester>


# Copyright 2000 až do data poslední změny Charles Curley.


# Další informace poskytne autor Charles Curley na
# http://www.charlescurley.com/.



# Název hostitele počítače, který má být zálohován.
target=tester
#zip=/mnt/zip
export zip="/var/bare.metal"; # Umístění archivů. Sem nepatří jednotka ZIP.


echo Zálohování počítače $target echo Snížení aktuálnosti záloh na disku ZIP. rm -r $target.old.zip mv $target.zip $target.old.zip # ssh $target "modprobe

ppa ; mount -r $zip" echo Kopírování disku ZIP. # -r pro rekurzivní kopírování, -p zachová časy a oprávnění, -q pro


# tichý režim bez ukazatele průběhu. scp -qpr $target:$zip $target.zip du -hs

$target.*zip


echo Snížení aktuálnosti archivů rm $target.tar.old.bz2
mv $target.tar.bz2 $target.tar.old.bz2

echo Vyčištění starých balíčků yum ssh $target "yum clean packages"

echo Zálohování počítače $target na zálohovací server.

# Přepínač "--anchored" zabraňuje tomu, aby přepínač --exclude vyloučil # všechny soubory s tímto názvem. Je např. vhodné vyloučit pouze
adresář /sys, # nikoli některé jiné položky sys v souborovém systému.

ssh $target "cd / ; tar -cf - --anchored --exclude sys --exclude $zip\ --exclude $zip.old --exclude mnt --exclude proc --exclude var/spool/squid\
  *" | bzip2 -9 | cat > $target.tar.bz2
```

```
# ssh $target "eject $zip"


echo Testování výsledků.
find . -iname "*.bz2" | xargs bunzip2 -t
```

restore.tester

```
#! /bin/sh

# Skript obnovuje všechna data na počítač tester přes ssh. Jedná se o poslední

# fázi obnovení.

# Časové razítko: <2003-04-24 09:59:45 ccurley restore.tester>

# Copyright 2000 až do data poslední změny Charles Curley.

# Další informace poskytne autor Charles Curley na



# http://www.charlescurley.com/.

# Název hostitele obnovovaného počítače.

target=tester

bunzip2 -dc $target.tar.bz2 | ssh $target "cd / ; tar -xpkf - "

ssh $target lilo
```

Zdroje informací

Nejsou uvedeny v konkrétním pořadí. Některé zdroje můžete prozkoumat sami. Tento seznam nelze považovat za doporučení. Ve skutečnosti jsem v mnoha případech produkt nepoužil a nemohu se k němu vyjádřit.

Network-booting Your Operating System (*http://osdev.berlios.de/netboot.html*) popisuje několik metod spouštění po síti pomocí zavaděče grub a některých dalších triků. Nezkou-šel jsem to, ale domnívám se, že s dobře připravenou disketou můžete do obnovovaného počítače dostat celý obraz první fáze.
Smart Boot Manager (SBM) (*http://btmgr.webframe.org/*) je plně funkční správce spouštění nezávislý na operačním systému, který má snadno použitelné uživatelské rozhraní. K dis-pozici je několik ukázkových obrazovek. „Neobejdete se bez něj, pokud váš systém BIOS neumožňuje spouštění z disku CD-ROM a chcete v první fázi obnovení použít systém Linux na disku CD-ROM.“
Vynikající kniha W. Curtise Prestona *Unix Backup & Recovery* (Zálohování a obnovení systému Unix). Díky této knize jsem se začal zabývat touto problematikou obnovení od základů. Rozhodně ji doporučuji.

Starý (2000) seznam malých distribucí systému Linux (*http://www.fokus.gmd.de/linux/linux-distrib-small.html*). Aktuálnější dokument v češtině najdete například na *http://cs. wikipedia.org/wiki/Seznam_distribuc%C3%AD_Linuxu.*
tomsrtbt (*http://www.toms.net/rb*). „Téměř celý systém Linux na jediné disketě.“ Tom také uvádí odkazy na jiné malé distribuce.
Projekt Linux Documentation Project (*http://www.tldp.org/*). Zejména odkazuji na doku-ment „*LILO, Linux Crash Rescue HOW-TO*“.
Nástroj parted nadace Free Software Foundation (*http://www.gnu.org/software/parted*) pro úpravy (zvětšení, zmenšení a přesunutí) oddílů.
QtParted (*http://qtparted.sourceforge.net/*) má nejspíš podobné funkce a grafické uživatel-ské rozhraní.
■ Partition Image (*http://www.partimage.org/*) k zálohování oddílů. Informace na webové stránce: „Partition Image je nástroj pro systém Linux/UNIX, který ukládá oddíly v mnoha formátech (viz dále) do souboru – obrazu oddílu. Obraz lze kom-primovat do formátů GZIP/BZIP2 kvůli úspoře místa na disku a rozdělit jej na více sou-borů, které je možné kopírovat na výměnné disky (například ZIP) ... Od verze 0.6.0 lze oddíly ukládat po síti.“
Bacula (*http://sourceforge.net/projects/bacula*) je zálohovací program uvolněný pod licencí GLP, který obsahuje kód pro obnovení od základů zčásti inspirovaný tímto návodem.

g4u (ghost for unix – *http://www.feyrer.de/g4u/*) je spouštěcí disketa či disk CD-ROM pro systém NetBSD, který umožňuje snadné klonování pevných disků počítačů PC při instalaci na mnoha počítačích pomocí FTP. Disketa nebo disk CD poskytuje dvě funkce: Jednak umožňuje odeslat komprimovaný obraz místního pevného disku na server FTP, jednak dovo-luje obnovit tento obraz pomocí FTP, dekomprimovat jej a zapsat jej zpět na disk. Konfigu-race sítě se získává ze serveru DHCP. Protože se s pevným diskem pracuje jako s obrazem, lze pomocí nástroje g4u obnovat (klonovat) libovolný souborový systém a operační systém.
Z webu *http://www.cs.utah.edu/flux/papers/frisbee-usenix03-base.html*: „Nabízíme nástroj Frisbee což je systém pro ukládání, přenos a instalaci obrazů celého disku. Jeho cílem je zvýšit rychlost a škálovatelnost v prostředí lokální sítě. Nástroj Frisbee mimo jiné používá vhodně upravenou metodu komprese zohledňující souborový systém, vlastní spolehlivý protokol typu multicast na aplikační úrovni a pružnou tvorbu rámců na aplikační úrovni. Výsledkem tohoto návrhu je systém, které dokáže rychle a spolehlivě distribuovat obraz disku mnoha klientům současně. Nástroj Frisbee dokáže například při použití standardní-ho hardwaru PC zapsat celkem 50 GB dat na 80 disků za 34 sekund. Popisujeme návrh a implementaci nástroje Frisbee, kontrolujeme důležitá rozhodnutí týkající se návrhu a hodnotíme jeho výkon.“

K dispozici je mnoho distribucí na přenosných discích USB. Podrobnosti naleznete na webu DistroWatch.
Záchranné sady založené na discích CD-ROM. Netvrdím, že tento seznam je vyčerpávající. Pokud o nějakém víte (nebo dokonce o takovém, který se za vyčerpávající vydává), dejte mi prosím vědět. Aktuálnější informace mohou být k dispozici na webu DistroWatch.
Mondo, jehož autorem je Hugo Rabson (*http://www.microwerks.net/~hugo/*), „...vytváří jeden nebo více spustitelných záchranných disků CD (nebo pásků a disket), které obsahují část vašeho souborového systému nebo celý systém. V případě katastrofické ztráty dat budete moci obnovit systém od základů...“.

- Sada s názvem The Crash Recovery Kit for Linux (*http://crashrecovery.org/*).
- „System recovery with Knoppix“ (Obnovení systému pomocí distribuce Knoppix – *http://www-106.ibm.com/developerworks/linux/library/l-knopx.html?ca=dgr-lnxw04Knop-pix*) představuje dobrý obecný úvod k obnovení systému a poskytuje některé užitečné odkazy související s distribucí Knoppix.

„Cool Linux CD (*http://emergencycd2.sourceforge.net/*) je živé CD se systémem Linux. Pou-žívá jádro řady 2.4 a některé bezplatné programy a demoverze.“
SystemRescueCd (*http://www.sysresccd.org/index.en.php*) „je systém Linux na spustitelném disku CD-ROM, který umožňuje opravit systém a data po havárii. Má také usnadnit úkoly správy počítače, jako například vytváření a úpravy oddílů na pevném disku. Obsahuje mnoho systémových nástrojů (parted, partimage, fstools...) a základní programy (edito-ry, midnight commander, síťové nástroje). Snaží se o co nejsnazší používání: Stačí spus-tit z disku CD-ROM a můžete dělat cokoli. Jádro systému podporuje nejdůležitější sou-borové systémy (ext2/ext3, reiserfs, xfs, jfs, vfat, ntfs, iso9660) a síťové protokoly (samba a nfs)“.
Syslinux (*http://syslinux.zytor.com/*) generuje spouštěcí kód pro diskety, disky CD-ROM a obrazy Intel PXE (Pre-Execution Environment). Nezávisí na obrazu diskety. Můžete vytvo-řit vlastní disky CD s mnoha nástroji, jako např. tomsrtbt.
Pokud byste chtěli vytvořit svůj vlastní systém: „Linux Live (*http://www.linux-live.org/*) je sada skriptů shellu bash, které umožňují vytvořit vaše vlastní živé CD z libovolné distribu-ce Linux. Stačí instalovat oblíbenou distribuci, odstranit všechny méně důležité soubory (například manuálové stránky a všechny další soubory, které nepotřebujete) a potom stáh-nout a spustit tyto skripty.“
„Disk CD PPART (*http://www.linbox.com/en/ppart.html*) dovoluje generovat spustitelný disk CD pro obnovení systému z dříve uložených pevných disků.“
Sada Rescue CD Set autora Timo (*http://rescuecd.sourceforge.net/*): „Tato sada představuje můj přístup, jak snadno generovat záchranný systém na spustitelném disku CD, který lze snadno přizpůsobit vašim požadavkům. Projekt se stále více vyvíjí směrem k systému ‚Debian na CD‘, takže kromě použití systému jako záchranného disku CD lze také instalo-vat celý systém Debian na CD.“
Seznam „List of Live CDs“ (*http://www.frozentech.com/content/livecd.php*) uvádí další live-CD distribuce.

Optimální použití písem v systému Linux

Úvod

Na pracovní ploše můžete mít nejúžasnější grafické téma, nejvychytanější kombinaci barev a na pozadí pracovní plochy nádherný

obrázek. Pracovní plocha však nebude vypadat profesionálně, zajímavě, čistě a komfortně bez použití dobrých písem.

V současnosti začíná obecně platit, že pěkně vypadající písma jsou klíčovým elementem dobré použitelnosti pracovní plochy, protože před počítačem obvykle strávíme několik hodin psaním dokumentů, prací s rozsáhlými tabulkami, tvorbou prezentací, procházením Internetu nebo komu-nikováním s přáteli. Celý den přitom čteme texty.

Subsystém písem v operačním systému Linux se v posledních letech značně vyvíjel, ze starého způsobu pojmenování a práce s písmy až po podporu písem typu TrueType, Bitstream Vera atd. Při uvedení systému Fedora Core 2 dosáhly komponenty Xft, FreeType a FontConfig a jejich vyu-žívání softwarem vyšších úrovní stabilní verze a v současnosti jsou považovány za dokončené. I přesto má Linux stále problémy s optimálním vykreslováním písem, přičemž většina těchto pro-blémů má vztah k softwarovým patentům, které budou popsány v další kapitole.

Proč není využívání písem v Linuxu dostatečně jednoduché?

Pokud chcete pouze překonfigurovat písma na pracovní ploše, přejděte na další kapitolu tohoto

dokumentu. Tuto kapitolu si přečtěte v případě, že chcete znát další podrobnosti. Písma se používají na obrazovce a pro tisk. Obě média se liší v rozlišení, měřeném v jednotkáchDPI. Obrazovky mají obvykle rozlišení od 72 do 96 DPI, moderní tiskárny používají až 1 200 DPI.Média s nízkým rozlišením, jako jsou obrazovky, proto potřebují kvalitnější vykreslovací algorit-my, pomocí kterých se obejde omezení daného média.
Aby se docílilo optimálního zobrazení písem na obrazovce, je potřeba:

■ Kvalitní písma, navržená pro médium s nízkým rozlišením. Technologie písem TrueType je v současnosti asi ta nejlepší možná využitelná technologie. Pro optimální zobrazení je však nutné použít písma, která byla navržena přímo pro tyto účely. Zjistili jsme, že nejlepší písma, která je možné zobrazit na obrazovce, jsou Tahoma a Verdana.
Kvalitní vykreslovací systém písem.

Aktuální distribuce Linuxu obsahují excelentní a vyspělou knihovnu pro vykreslování písem s názvem FreeType.

Soubor .ttf obsahuje informace pro vykreslení znaků v libovolné velikosti, takže s využitím nástrojů OpenOffice.org nebo CorelDraw je možné konvertovat text do grafické podoby (sesta-vené z čar a kvadratických beziérových křivek).

Algoritmy vykreslování písem jsou extrémně složité, protože musí rozhodnout o vybarvení jedno-tlivých pixelů na základě matematických výrazů, uložených v souboru .ttf. Pokud používáte text
o velikosti 48 nebo 60 a algoritmus jeden nebo dva pixely nevykreslí, není to taková tragédie. Pokud je text nutné zobrazit písmem o velikosti 8 nebo 11 bodů, záleží na každém pixelu. Tyto velikosti textu se používají v prostředích KDE nebo Gnome pro zobrazení textu při procházení Internetu a pro zobrazení téměř všeho, co je na obrazovce. Pro efektivní řešení tohoto problému kromě matematických výrazů, které jsou součástí souboru .ttf, návrhář písem (člověk, který pou-žívá software pro tvorbu písem) vkládá další dodatečné informace, které pomáhají vykreslovací-mu systému provádět správná rozhodnutí při vykreslování písem malých velikostí. Dodatečné informace a proces jejich tvorby se nazývá *grid-fitting* nebo *hinting*.

Faktem je, že technologie, které napomáhají tyto dodatečné informace interpretovat, jsou patentovány firmou Apple a obecně se nazývají interprety bajtového kódu TrueType (TrueType Byte Code Interpreters – dále jen BCI).

Při implementaci projektu FreeType bylo využito reverzního engineeringu a jeho výsledkem byl interpreter bajtového kódu. Díky problémům s legálností v některých zemích je tento vykreslova-cí systém při překladu a vytváření balíčků deaktivován. V následující tabulce naleznete seznam distribucí i se stavem využívání tohoto vykreslovacího systému. Pokud máte informace pro aktu-alizaci tohoto seznamu, prosím zašlete nám je.

Stav interpretu bajtového kódu v jednotlivých distribucích

Podporuje Nepodporuje

Conectiva Mandriva Tentok Red Hat, Fedora

Vykreslovací systém FreeType se pokouší obejít problémy s legálností vykreslovacích algoritmů, ale podle našich testů algoritmy BCI dávají při zobrazování na obrazovce daleko lepší výsledky. Taktéž platí, že pro většinu distribucí, které původní BCI neobsahují, najdete balíčky s jeho pod-porou v neoficiálních repozitářích.

Subsystémy písem X.org

V současnosti X.org a XFree86 využívají dva subsystémy písem, každý s jinými charakteristikami:

Původní (více než 15 let starý) subsystém má název „*core X font subsystem*“ (XFS). Písma vykreslovaná tímto subsystémem nemají vyhlazované hrany, obsluhuje je X server a mají názvy podobné tomuto:*-misc-fixed-medium-r-normal—10-100-75-75-c-60-*

iso8859-1.
Nový subsystém má název *„fontconfig“* a umožňuje aplikacím přímý přístup k souborům písem. Fontconfig se obvykle využívá v součinnosti s knihovnou Xft, která umožňuje apli-kacím vykreslovat písma na obrazovce s využitím vyhlazování hran. Fontconfig používá čitelnější názvy písem, například: *Luxi Sans-10*.

Subsystém Fontconfig/Xft postupem času nahradí starší subsystém XFS. V současnosti využívají Fontconfig/Xft aplikace, které jsou vytvořeny pomocí knihoven Qt3 nebo GTK 2 (to zahrnuje i aplikace pro KDE a GNOME). Ostatní aplikace využívají subsystém X font. V budoucnu budou moci distribuce Linuxu podporovat pouze subsystém Fontconfig/Xft místo serveru písem XFS, který je v současnosti výchozím způsobem pro přístup k písmům.

> Poznámka
>
> Výjimkou v používání subsystému písem, uvedeného výše, je OpenOffice.org, který využívá vlastní technologii vykreslování písem.

Snadný postup pro zlepšení vzhledu pracovní plochy

Bude nutné provést následující akce:

Aktualizace balíčku knihoven FreeType balíčkem přeloženým s podporou pro BCI.
Instalace balíčku Webcore Fonts (známého také pod názvem Microsoft Fonts).
Dodržení instrukcí uvedených v této kapitole, které popisují, jak provést konfiguraci pra-covní plochy a běžných aplikací.

Získání lepšího balíčku RPM s podporou FreeType

Balíček FreeType přeložený s podporou pro BCI má daleko lepší výsledky při zobrazování písem
na obrazovce.
Balíčky RPM pro jednotlivé distribuce naleznete zde:

CentOS, Red Hat Enterprise Linux 3 a 4, Fedora 3, 4, 5 a 6: *http://avi.alkalay.net/software/ freetype.bci/FC3-FC4-RHEL3-RHEL4/*.
Fedora 5: *http://avi.alkalay.net/software/freetype.bci/FC5/*. Autorem je Cody DeHaan.
Mandriva: Pomocí webových stránek Penguin Liberation Font na adrese *http://plf.zarb.org/*. Název balíčku je *libfreetype6*.
Uživatelé distribuce Debian Sarge mohou využít balíčky FreeType s podporu BCI z úložišť „testing“ a „unstable“. Další stabilní distribuce Debianu bude tyto balíčky využívat implicit-ně. Název balíčku je v tomto případě *libfreetype6*.

Pokud využíváte některou z uvedených distribucí na platformě, pro kterou nejsou balíčky RPMposkytovány, můžete si snadno přeložit vlastní verze (dokonce i v případě, kdy s překlady soft-waru nemáte žádné zkušenosti) s využitím popisu, který naleznete v dodatku B.

Oceníme, pokud přispějete vlastními informacemi o tvorbě balíčků pro specifické distribuce. Kon

taktujte nás na adrese <avi na unix.sh>. Pokud se zajímáte o vytvoření vlastního balíčku FreeType, můžete jako výchozí bod využít doda-tek A, který obsahuje informace o vytvoření balíčku FreeType s podporou BCI pro distribuceFedora Core a Red Hat.

Konfigurace pracovní plochy

Základní myšlenkou je použít všude písmo, které obsahuje kvalitní dodatečné informace (hinting). V našem případě využijeme pro uživatelské rozhraní písmo Tahoma 8 bodů, pro neproporcionální text písmo LucidaTypewriter 8 bodů a pro čtení textů nebo prohlížení Internetu písmo Verdana 8, 9 nebo 10 bodů. Uvedené velikosti jsou výchozí velkosti písem na pracovní ploše systému Micro-soft Windows a vypadají pěkně při rozlišení 1 024 x 768 bodů. Pokud používáte vyšší rozlišení obra-zovky (1 280 x 1 024 nebo 1 600 x 1 200), je naše doporučení použít stejná písma o větší velikosti.

Tato písma (a hlavně Tahoma a Verdana od firmy Microsoft) byla vybrána proto, že vypadají per-fektně i v malých velikostech (8 bodů) a umožňují efektivní využití obrazovky. Pracovní plocha při použití těchto písem vypadá pěkně, profesionálně a komfortně. Písma byla vytvořena s ohle-dem na tyto účely.

Pro názvy oken nebo text, který bude zobrazen většími velikostmi, je možné zvolit jakékoliv jiné
písmo, protože u větších velikostí nejsou dodatečné informace pro vykreslování tak důležité. Většina moderních distribucí Linuxu jako openSUSE nebo Mandriva obsahují lokalizovaná písma DejaVu – vycházející z písem Bitstreeam Vera zmiňovaných dále – která se podle některých názo-rů kvalitou přibližují písmům Microsoftu. Vyzkoušejte a uvidíte, možná s nimi vystačíte.

Poznámka na téma vyhlazování hran (anti-aliasing)

Vyhlazování hran je technologie, která snižuje „kostrbatost“ při zobrazování písem na médiích s nízkým rozlišením a používá se pro zvýšení kvality zobrazování textu na obrazovce. Také se využívá pro maskování nedostatků u písem, která mají špatně navržené dodatkové informace při zobrazení písma menšími velikostmi.

Praktický poznatek, který z používání vyhlazování hran vzešel, říká, že vyhlazování hran je vhod-né používat u velikostí písem větších než 10 bodů a pro menší velikosti textu použít raději dobře navržená písma bez použití vyhlazování hran. Zatím nejlepší písma, která je možné najít, jsou písma z balíčku Webcore Fonts.

KDE

Při konfiguraci prostředí KDE použijte Control Centrum (v českém prostředí Ovládací centrum

nebo příkaz kcontrol v příkazovém řádku). Konfigurace vypadá následovně:Pro titulky oken bylo zvoleno písmo Trebuchet 12 bodů, pro neproporcionální text písmo Luci-daTypewriter 8 bodů a pro vše ostatní (tj. nabídky, tlačítka atd.) písmo Tahoma 8 bodů (v českémpřekladu 9 bodů). První dvě písma můžete nakonfigurovat dle vlastní libosti, ale použití písmaTahoma 8 bodů je nejoptimálnější varianta, která je použitá i v prostředí systémů Microsoft Win-dows 2000 a XP.

V konfiguraci bylo dále zakázáno vyhlazování hran písem pro písma do velikosti 9 bodů. Pro-hlédněte si zobrazení dialogového okna a všimněte si, jak je text jasně vykreslen, vypadá čistěa profesionálně.

Prohlížeč Konqueror (prohlížeč a správce souborů v prostředí KDE) také vyžaduje konfiguraci

písem.Pro zobrazení seznamu souborů v okně prohlížeče Konqueror je použito písmo Tahoma 8 bodů,protože tento typ písma byl navržen s ohledem na použitý účel. Velikost 8 bodů je nejdůležitějšívelikost a i bez použití vyhlazování hran písmo vypadá čistě a pěkně.
A jako poslední je konfigurace pro procházení Internetu. Jako základní písmo se využívá Verda-na, protože bylo navrženo pro účely přehledného čtení textu na obrazovce. Jako neproporcionální písmo byl zvolen typ LucidaTypewriter. Jako náhradu tohoto písma je možné zvolit například písma Courier nebo Bitstream Vera Mono.

Ostatní písma zůstanou v konfiguraci nenastavená. Jako patkové písmo je možné použít písmo Times New Roman. Více informací o patkových písmech naleznete v kapitole „Patková nebo nepatková“.

Velikost písem pro zobrazení webových stránek je individuální a závisí kromě jiného i na zdraví očí uživatele a na rozlišení obrazovky. Při velikosti 1 024 x 768 je optimální využít velikost písma 8 bodů, ale rozhodně ne menší než 7 bodů. Nastavení konkrétní velikosti však není tak účinné, protože moderní webové stránky velikosti použitých písem nastaví na absolutní hodnoty. Proto je daleko praktičtější použít nabídku „Zobrazit“ prohlížeče a použít zvětšení nebo zmenšení celé aktuálně zobrazené webové stránky.

Poslední poznámka bude věnována výchozímu kódování. To je téma, které leží za hranicí rozsahu tohoto dokumentu, nicméně je vhodné nastavit kódování podle nastaveného jazyka. Nastavení je možné změnit v případě, kdy často procházíte stránky, které neobsahují text v čisté sadě ASCII (mezinárodní text), vytvořené autory webu, kteří se ještě nedoslechli o kódování UTF-8. I v tomto případě je pak praktičtější použít nabídku „Zobrazit“ prohlížeče a nastavit aktuální kódování přímo pro aktuálně zobrazenou webovou stránku.

Gnome

Obecná pravidla použijeme i zde: písmo Tahoma 8 bodů pro vše.

OpenOffice.org

Od uvedení systému Fedora Core 3 má OpenOffice.org vzhled integrovaný s KDE a Gnome (týká se to i jiných distribucí, například Mandriva Linuxu nebo openSUSE). To znamená, že prostředí operačního systému by mělo být schopno říci aplikacím OOo, jak využívat písma. To však ve skutečnosti nefunguje. Dalším pátráním bylo zjištěno, že do prostředí OOo se nepřenesly pouze ty konfigurace, které nevyužívaly vyhlazování hran písem. Tento stav je možné změnit přímo v prostředí OOo

V konfiguračních nastaveních bylo aktivováno vyhlazování hran písem pro písma o velikosti 12 pixelů (což je cca 9 bodů) a výsledek vidíte sami: Čisté a komfortní ovládací prvky, které využívají písmo Tahoma 8 bodů.

Mozilla Firefox

Prohlížeč Mozilla Firefox nakonfigurujeme podobně jako Konqueror. Po výběru položek nabídky Úpravy -> Předvolby, pak Obsah a tlačítko Rozšířené v části Písma a Barvy. Pro procházení webu vybereme písmo Verdana 14 bodů a pro neproporcionální text LucidaTypewri-ter 11 bodů. Firefox je aplikace, která využívá Gnome, takže využije nastavení písem Gnome pro ovlá-dací prvky. Velmi zajímavý způsob konfigurace vykreslování písem v prostředí Firefox je popsa-ný na webových stránkách Mandriva Wiki na adrese *http://wiki.mandriva.com/en/Docs/Desk-top/Browsers/Firefox.*

Pěkné alternativy bez použití písem Webcore fonts.

Pokud se chcete vyhnout použití patentů nebo proprietárních písem, je nejlepším způsobem pou-žít písma Bitstream Vera Sans 8 bodů (v českém vydání bylo použito počeštěné písmo DejaVu),Nimbus Sans 8 nebo 9 bodů nebo Luxi Sans 8 nebo 9 bodů pro ovládací prvky a větší velikostipro text. U těchto písem je nutné použít vyhlazování hran, aby se eliminovala menší kvalita doda-tečných informací při vykreslování těchto písem.
Následují ukázky použití těchto písem v prostředí KDE. Sledujte pozorně, jak jsou vykresleny popisky ovládacích prvků.
Jak sami vidíte, výsledky nejsou tak pěkné jako při použití písma Tahoma 8 bodů.

Balíčky písem

Písma Bitstream Vera

Bitstream přinesl do světa open source množinu písem Vera, která jsou kvalitní a obsahují patko-vá, bezpatková i neproporcionální písma. Písma neobsahují kvalitní dodatečné informace pro vykreslování, ale dají se použít pro ovládací prvky, programování, plynulé čtení textu a prohlíže-ní webových stránek. Následuje ukázka obrazovky.
U větších velikostí tato písma vypadají perfektně, zvláště při použití vyhlazování hran. Zhoršené kvality vykreslování je možné zaznamenat pouze u malých velikostí. Písma Bitstream Vera jsou součástí všech moderních distribucí Linuxu.

Písma Webcore Fonts

Oficiální distribuce těchto písem pro Linux, kterou naleznete na adrese *http://avi.alkalay.net/ software/webcore-fonts*, obsahuje archivy tarball a balíčky RPM pro různé distribuce. Tato písma jsou také známá pod názvem Microsoft Fonts a jsou to ta nejlepší písma, která je možné na obra-zovce použít. Obsahují kvalitní dodatečné informace pro vykreslování, takže i v malých velikos-tech jsou velmi vhodná pro ovládací prvky i text.

Součástí tohoto balíčku písem jsou Verdana, Tahoma, Times New Roman, Trebuchet, Comic Sans, Impact a ostatní. Ukázka obrazovky viz následující strana. Jak bylo zmíněno dříve, písma Tahoma a další písma jsou navržena pro použití na obrazovce, ale obvykle se využívají i pro jiné účely. Cílem této kapitoly je poskytnout odkazy, na kterých nalez-nete kvalitní balíčky RPM nebo DEB právě pro vaši distribuci. Balíčky poskytují nezávislí přispě-vatelé, a pokud máte možnosti sestavit tyto balíčky pro vaši distribuci, kontaktujte nás prosím a zašlete adresu, na které vaše balíčky nalezneme.
Balíčky pro distribuce:

Red Hat a Fedora (viz *http://avi.alkalay.net/software/webcore-fonts*). Toto je původní balí-ček a pracuje i v prostředí jiných distribucí.
Balíčky pro starší verze Mandrake Linuxu (viz *http://rpm.borgnet.us/10.1/media/RPMS/noarch/*) od Scotta Graybana <sgrayban na borgnet.us>.
Dále je možné vyrobit RPM balíčky z *http://corefonts.sourceforge.net/* (fungují např. v Man-driva Linuxu).
Prosím, zašlete nám balíčky například pro Slackware a Debian.

Po nainstalování těchto balíčků písem si určitě všimnete lepšího vykreslování webových stránek, protože tato písma používají i profesionální návrháři webových stránek. Někteří tvrdí, že tato písma jsou pro volné použití pouze pro ty, kteří vlastní licence pro systém Microsoft Windows. Podle původní licence fontů se ale zdá, že je možná volná redistribuce v nezměněné formě, viz původní znění na *http://corefonts.sourceforge.net/eula.htm*.

Tvorba přenositelných dokumentů

Určitě se vám již stalo, že jste vytvořili perfektní dokumenty, prezentace, tabulky nebo webové stránky, které na vašem počítači vypadají perfektně, ale po otevření na počítači kolegy vypadají, jako by se veškeré formátování ztratilo. V této kapitole naleznete informace, které pomohou takovým problémům zabránit.

Z Linuxu do Windows a naopak

Pokud je nutné vyměňovat dokumenty s uživateli systému Windows, je vhodné používat písma Windows. To je obecné pravidlo. V tomto případě je vhodné nainstalovat balíček Webcore Fonts a v dokumentech používat pouze písma Arial, Times New Roman, Verdana atd.

Kombinace těchto písem se systémem určeným pro více platforem, jako je OpenOffice.org, dává dohromady skutečně produktivní nástroj pro týmovou spolupráci.

Z Linuxu na Linux

Písma, pomocí kterých je možné vytvořit dobře vypadající dokumenty v moderních distribucích Linuxu, jsou uvedena v tabulce 6.2.

Písma v Linuxu

Písmo

Bitstram Charter Rodina písem Bitstream Vera Century Schoolbook Rodina písem Luxi Rodina písem Nimbus URW Palladio URW Bookman URW Chancery URW Gothic Utopia

Pomocí těchto písem budete schopni bezpečně vyměňovat a tisknout dokumenty v prostředí růz-ných distribucí systému Linux. Existují i jiná písma v systémech Linux, ale nejsou zde uvedena, pro-tože jsou to většinou málo kvalitní rastrová písma pro použití na obrazovce a ne v dokumentech.

Výměna mezi libovolnými systémy s využitím OpenOffice.org a písem Bitstream Vera

Nadpis říká vše. Systém OpenOffice.org obsahuje ve všech distribucích pro různé platformy balí-ček písem Bitstream Vera. Pokud budete používat pouze tato písma, dokumenty budou vypadat stejně ve všech instalacích systému OpenOffice.org. Jako poznámku pod čarou bychom rádi uved-li, že OpenOffice.org exceluje v přenositelnosti. Systém OOo se chová a vypadá na všech plat-formách stejně a to platí i o vzhledu dokumentů. Je to prostě skvělý nástroj.

Malý průvodce styly

Aby vaše dokumenty měly profesionální vzhled, je vhodné písmo vybrat s ohledem na účel vytvá-řeného dokumentu. V současnosti je standardem patkové písmo (Times atd.), které je vhodnouvolbou pro knihy a časopisy. Kromě toho si svůj prostor dobývají i nepatková písma, podle názo-ru mnohých uživatelů mají právě díky chybějícím patkám daleko modernější vzhled. Někdy sevyužívají v tištěných článcích a komerčních dokumentech.
Pro webové stránky jsou tou správnou volbou písma Arial, Helvetica a obzvláště Verdana.Více podrobnějších informací naleznete v kapitole „Průvodce návrháře moderních dobře vypada-jících dokumentů“, kterou napsal Donovan Rebbechi a která se zabývá kulturními i sociálnímivlivy, jež ovlivnily vývoj návrhů písem, a také tím, co vytvářejí návrháři v současnosti.

Vytvoření balíčků RPM s písmy

Nahodilé ukládání souborů .ttf není ten nejlepší způsob práce s písmy. Ztěžuje migraci a po čase zaneřádí počítač. Software pro správu balíčků, například RPM, umožní písma snadno insta-lovat standardním způsobem, umožní správu aktualizací písem a výrazně usnadní rozsáhlou distri-buci a hromadné nasazení písem. V této kapitole naleznete informace o tom, jak snadno vytvořit balíčky RPM s vybranými písmy. Budeme rádi, pokud přispějete svými vlastními zkušenostmi z vytváření jiných typů balíčků. Vyhovují-li vám hotové balíčky z uvedených adres, není třeba se jejich výrobou dále zabývat.

Krok 1: Příprava prostředí pro vytvoření balíčku

Pro vytváření balíčků RPM je nutné vytvořit speciální strukturu složek a příslušným způsobem nakonfigurovat prostředí. Všechny kroky je možné provádět pod účtem běžného uživatele. Fakticky nedoporučujeme tyto kroky neprovádět pod účtem uživatele root.

Pro vytvoření všech složek je nutné provést následující příkazy:

```
bash$ cd ~bash$ mkdir -p src/rpmbash$ cd src/rpmbash$ cp -r /usr/src/redhat/* .bash$ lsBUILD/ RPMS/ SOURCES/ SPECS/ SRPMS/bash$
```

(Znak tilda ~ je zástupným znakem pro domovskou složku aktuálního uživatele a interpret píka-zů ví, jak použití tohoto zástupného znaku interpretovat.) Tyto příkazy provádíme na systému Red Hat, nicméně to nejdůležitější je, aby ve složce src/rpm existovaly následující podsložky:

```
BUILD/
RPMS/noarch/
SRPMS/
```

Dále je nutné v domovské složce vytvořit soubor .rpmmacros, který obsahuje jediný řádek:

```
%_topdir domovská_složka/src/rpm
```

Text domovská_složka je nutné nahradit absolutní cestou k domovské složce z proměnné $HOME. Soubor .rpmmacros může vypadat například takto:

```
%_topdir /home/aviram/src/rpm
```

Krok 2: Příprava souborů písem do balíčku

Teď je vhodné vymyslet název pro novou kolekci písem. V našem příkladu použijeme název *myfonts*. Pak je nutné vytvořit složku

~/src/myfonts/myfonts. Ano, název myfonts je dvakrát za sebou. Pak je nutné do této složky zkopírovat všechny potřebné soubory .ttf. Složky budou mít následující obsah:

bash$ cd ~/srcbash$ find myfonts/myfonts/

myfonts/myfonts/
myfonts/myfonts/font1.ttf

myfonts/myfonts/font2.ttf myfonts/myfonts/font3.ttf ...

Krok 3: Vytvoření souboru .spec se šablonou

Pro vytvoření balíčku RPM je nutné vytvořit soubor .spec, který obsahuje instrukce pro tvůrce balíčku s informacemi o tom, jak organizovat soubory, s popisem balíčku, jménem autora, copy-right atd. Vzorovou šablonu naleznete na adrese *http://tldp.org/HOWTO/Font-HOWTO/rpm.html.* Šablona vypadá takto:

Šablona souboru .spec

Name: myfonts [1]Summary: Kolekce legračních písem [2]Version: 1.0 [3]Release: 1License: GPL [4]Group: User Interface/XSource: %{name}.tar.gzBuildRoot: %{_tmppath}/build-root-%{name}BuildArch: noarchRequires: freetypePackager: Avi Alkalay <avi unix sh> [5]Prefix: /usr/share/fontsUrl: http://myfonts.com/ [6]

%description [7]
Toto jsou písma, použitá při marketingové kampani, vytvořená speciálně pro nás
naší marketingovou agenturou.
Balíček obsahuje následující písma: Bodoni, Bodoni Black, Company Logo, Outline
Company Logo atd.

%prep

%setup -q -n %{name}

%build

%install
mkdir -p $RPM_BUILD_ROOT/%{prefix}
cp -r %{name}/ $RPM_BUILD_ROOT/%{prefix}

%clean
rm -rf $RPM_BUILD_ROOT

%files
%defattr(-,root,root,0755)
%{prefix}/%{name}

%post { ttmkfdir -d %
{prefix}/%{name} \
-o %{prefix}/%{name}/fonts.scaleumask 133/usr/X11R6/bin/mkfontdir %{prefix}/%{name}/usr/sbin/chkfontpath -q -a %
{prefix}/%{name}[-x /usr/bin/fc-cache] && /usr/bin/fc-cache
} &> /dev/null || :

%preun {
if [“$1” = “0”]; then
cd %{prefix}/%{name}

```
                rm -f fonts.dir fonts.scale fonts.cache*

        fi } &> /dev/null || :

%postun if [ "$1" = "0" ]; then
        /usr/sbin/chkfontpath -q -r %{prefix}/%{name} fi [ -x /usr/bin/fc-cache ] && /usr/bin/fc-cache



%changelog [8]
* Čtvrtek, 14 prosince 2002 Avi Alkalay <avi unix sh> 1.0
-testováno
-Připraveno pro použití
* Pondělí, 10 prosince 2002 Avi Alkalay <avi unix sh> 0.9
-První verze šablony
```

V šabloně je nutné odpovídajícím způsobem změnit následující položky:

[1] Název balíčku kolekce písem.

[2] Stručný popis obsahu balíčku.

[3] Verze balíčku.

[4] Licence pro použití balíčku.

[5] Jméno autora, který balíček vytvořil.

[6] Adresa, na které jsou uvedeny další informace o balíčku. Pokud taková adresa není, řádek je možné vypustit.

[7] Podrobný popis jednotlivých písem.

[8] Vývoj a historie balíčku. Musí dodržovat formát, použitý v příkladu.

Soubor musí mít stejný název, jako je název balíčku, v našem případě je to myfonts.spec. Sou-bor musí být uložen v hlavní složce balíčku. Celá struktura souborů a složek bude nakonec vypa-dat takto:

```
bash$ cd ~/src bash$ find myfonts myfonts/
myfonts/myfonts.spec myfonts/myfonts/
myfonts/myfonts/font1.ttf myfonts/myfonts/font2.ttf
myfonts/myfonts/font3.ttf ...
```

Krok 4: Vytvoření balíčku

Balíček je už skoro hotov. Dále je nutné provést tyto příkazy:

```
bash$ cd ~/src bash$ tar -czvf myfonts.tar.gz myfonts bash$ rpmbuild -ta myfonts.tar.gz
```

A je hotovo (poté co zhlédnete velký počet zpráv o průběhu vytváření balíčku). V podstatě byl vytvořen soubor .tar.gz, který obsahuje všechny soubory písem a soubor myfonts.spec ana tento soubor byl použit nástroj rpmbuild, který v archivu vyhledal soubor myfonts.spec apro-vedl instrukce v něm obsažené.

Vygenerovaný balíček RPM naleznete ve složce ~/src/rpm/RPMS/noarch/. Ve složce ~/src/-rpm/SRPMS/ naleznete zdrojový soubor balíčku RPM, který je vhodné zálohovat a pak případně použít pro opětovné vygenerování balíčku RPM. Pokud je nutné balíček vygenerovat znovu, stačí použít příkaz:

```
bash$ rpmbuild -rebuild myfonts-1.0-1.src.rpm
```

Více informací o vytváření balíčků RPM získáte v knize Maximum RPM, kterou naleznete v mnoha formátech na adrese *http://www.rpm.org*.

Průvodce návrháře moderních dobře vypadajících dokumentů

V této kapitole naleznete informace o základech typografie. Nejsou to až tak důležité informace, přesto mohou být pro milovníky písem zajímavé. Další informace můžete najít například v knize LaTeX pro začátečníky (Konvoj, 80-7302-049-1) nebo přímo vynikající Typografický manuál (FONT, *http://www.font.cz/font/typoman.html*).

Klasifikace rodin písma Pevná versus proměnná

šířka

Rodiny písem se dají rozlišovat podle řady kritérií. Prvním z nich jsou písma s pevnou šířkou (neproporcionální) a písma s proměnnou šířkou (proporcionální). Písma s pevnou šířkou vypa-dají jako psaná na stroji, protože všechny znaky jsou stejně široké. Tento typ písma je vhodný např. pro textové editory nebo počítačovou konzolu, nehodí se ale pro delší bloky textu. Druhým typem jsou písma s proměnnou šířkou. Většinou používáme právě písma s proměnnou šířkou, i když užitečná jsou i ta s pevnou šířkou (v tomto dokumentu je používáme v příkladech příka-zů). Nejznámějším písmem s pevnou šířkou je Courier.

Patková nebo bezpatková

Patky jsou malé plošky na koncích znaků. Například písmeno „i“ v písmech, jako je Times Roman, má patky vedoucí ze základny „i“ a z vrcholku „i“. Patková písma jsou obvykle považována za lépe čitelná než bezpatková písma. Existuje jich celá řada typů. Bezpatková písma nemají ony plošky navíc a vypadají tak „ostřeji“. Obvykle se moc nehodí pro sazbu dlouhých dokumentů. Dobře však poslouží v dokumentech, které jsou určeny jen k rychlému prohlédnutí (například webové stránky, katalogy, reklamní brožury). Další vhodné použití mají jako obrazovková písma, zejména u malého písma. Díky menší míře detailů jsou za těchto okolností lépe čitelné. Microsoft například prohlašuje písmo Verdana za dobře čitelné i ve velmi malé velikosti.

Mezi významná bezpatková písma patří Lucida Sans, MS Comic Sans, Verdana, Myriad, Avant Garde, Arial, Century Gothic a Helvetica. Mimochodem, řada typografů považuje písmo Helvetica za nepříjemné. Používá se příliš často a v řadě knih o typografii se doporučuje tomuto písmu vyhnout.

Nové a staré – různé typy patkových písem

Medievalové písmo

Medievalová písma jsou založena na velmi tradičních stylech pocházejících až z 15. století. Tato písma mají velmi konzervativní vzhled a jsou dobře čitelná. Hodí se k sazbě delších dokumentů. Anglickým termínem „starý styl“ se míní styl, nikoliv stáří. Existují klasická medievalová písma, například Goudy Old Style, navržená ve 20. století. Tato skupina písem má následující vlastnosti:

Dobře definované výrazné patky.
Diagonální zvýraznění. Představte si písmo psané plnicím perem, v němž čáry pod úhlem 45 stupňů od vertikály proti směru hodinových ručiček jsou silnější a čáry pod úhlem 45 stupňů od vertikály po směru hodinových ručiček jsou tenčí. Medievalová písma tento efekt často používají.
Čitelnost. Medievalová písma jsou prakticky vždy velmi dobře čitelná.
Jemnost a nevelký kontrast. Písma používají širší a tenčí čáry, rozdíl mezi nimi je ale nepříliš velký.

Mezi známá medievalová písma patří Garamond, Goudy Old Style, Jenson a Caslon (to poslední je sporné, někdy bývá považováno za tranzitivní).

Moderní písmo

Moderní písma jsou protějškem medievalových. Mají typicky výraznější charakter a vzhled a pou-žívají se spíše ke zvýraznění částí dokumentu než k sazbě dlouhého textu. Nic ale není jen černé nebo bílé – některá moderní písma, například Computer Modern, Monotype Modern nebo New Century Schoolbook, jsou velmi dobře čitelná (kontrast mezi silnějšími a slabšími čarami je sníže-ný, a tak mají lepší čitelnost). Tato písma jsou založena na návrzích oblíbených v 19. století a později. Mezi jejich hlavní rysy patří:

Jemnější patky, často jen tenké vodorovné linky.
Svislé zvýraznění. Svislé čáry jsou silnější, vodorovné slabší.
Často výrazný kontrast mezi silnějšími a slabšími čarami.
Písma s větším kontrastem mezi silnějšími a slabšími čarami jsou hůře čitelná.

Asi nejznámějším písmem této kategorie je Bodoni. Dalšími jsou Computer Modern a Monotype Modern (z nějž předchozí zmíněné vychází).

Tranzitivní písmo

Tranzitivní písma patří někam mezi moderní a medievalová. Řada z nich je stejně dobře čitelná jako medievalová, vycházejí však z poněkud jiného designu. I když je na nich vidět přechod k modernímu stylu, jsou výrazně jemnější než klasická moderní písma. Do této kategorie patří například Times Roman, Utopia, Bulmer nebo Baskerville. Z nich Times má blíže k medievalové skupině, Bulmer pak blíže k moderní.

Písmo s plochou patkou

Písma s plochou patkou odvozují svůj název od tlusté, jakoby blokové patky, na rozdíl od jem-ných oblouků medievalových písem a tenkých linek moderních písem. Tato písma vypadají robustně a jsou poměrně dobře čitelná. Řada z nich má egyptské názvy – například Nile nebo Egyptienne (i když s Egyptem nemají nic společného). Tato písma jsou vynikající pro psaní textů, u kterých může utrpět kvalita (například texty určené ke kopírování, text v novinách). Nejznámější jsou Claredon, Memphis a Egyptienne, společně s několika písmy používanými v psacích strojích. Řada písem s plochou patkou má pevnou šířku a obráceně

– prakticky všechna písma s pevnou šířkou mají plochou patku.

Revoluce bezpatkových písem

Nástup bezpatkových písem je poměrně nový fenomén. První známější bezpatková písma byla navržena koncem 19. a začátkem 20. století. Mezi první patří Futura, Grotesque a Gill Sans. Tato písma představují reprezentanty „geometrické“, „groteskní“ a „humanistické“ skupiny bezpatkových písem.

Groteskní

Tato písma si vysloužila svůj název tím, že veřejnost byla zprvu poněkud šokována jejich poměr-ně výrazným vzhledem. Tato písma vypadají velmi prostě, protože nemají patky a jejich design je jednoduchý a jasný. Hodí se dobře pro sazbu titulků. Lépe čitelné varianty se používají v komik-sech a v marketingových brožurách, kde se text vyskytuje v menších celcích. Nevypadají tak umě-lecky jako geometrická písma. V porovnání s nimi mají větší variace v šířce, více tahů a jsou hra-natější (nepoužívají tolik kruhových oblouků). Významný rozdíl proti geometrickým písmům vyka-zují velké G a malé a. I zde jsou tato písmena jednoduchá, ale nezacházejí do takového extrému jako „brutálně“ avantgardní geometrická písma.

Mezi známé představitele této kategorie patří Helvetica, Grotesque, Arial, Franklin Gothic a Univers.

Geometrická

Písmo Futura bylo uvedeno sloganem form follows function – tedy forma slouží obsahu. Geome-trická skupina písem má výrazně minimalistický vzhled. Významným rysem je konstantní tloušťka linky. Je to zřetelné zejména u tučného řezu. Groteskní a humanistická písma v tučném řezu vyka-zují variace v tloušťce linky, u geometrických písem to nastává jen výjimečně. Další významný rys je minimalistický návrh. Znaky jsou téměř vždy tvořeny svislými a vodorovnými čarami v kombi-naci s kruhovými oblouky (vypadají jako kreslená pravítkem a kružítkem). Znaky obsahují mini-mum tahů. Nejvýrazněji se geometrická písma poznají podle velkého G, které se skládá jen ze dvou křivek – dlouhého kruhového oblouku a vodorovné čárky. Podobně výrazné je malé a – svislá čára a kružnice. Klasická podoba písmene „a“ se nepoužívá – je příliš složitá. Známá písma této kategorie jsou Avant Garde, Futura a Century Gothic.

Humanistická

Jak název možná napovídá, návrh těchto písem se snaží omezit mechaničnost jejich vzhledu. V řadě ohledů se podobají spíše patkovým písmům. Vypadají jako „rukou psaná“. Vykazují lehké variace tloušťky linky, což vyniká zejména v tučné variantě. Oblé části nejsou tak rigidní jako u geometrických písem. Mnohá používají dvoudílné malé g, podobného tvaru jako u klasických patkových písem. Dají se snáze používat, aniž by vzniklé dokumenty byly příliš ošklivé, protože jsou částečně příbuzná s klasickými patkovými písmy.

Kompatibilní písma

Kombinování různých písem není jednoduché, proto se doporučuje to s kombinacemi nepřehá-nět. Logická volba dvou druhů písem padá na patkové a bezpatkové. Lze říci, že dobře vypadá kombinace medievalových a humanistických písem anebo moderních a geometrických. Tranzitiv-ní písma se rovněž kombinují s humanistickými. Písma s plochou patkou se obvykle kombinují s groteskními písmy, i když některá z nich lze kombinovat i s geometrickými a humanistickými.

Vyplývá z toho tedy, že rozumná filozofie je kombinovat konzervativní patková písma a mírnější bezpatková a divočejší moderní patková písma s avantgardněji vyhlížejícími geometrickými.

Ligatury, kapitálky a speciální písma Ligatury

Volba vhodných rozestupů mezi písmeny s sebou přináší řadu problémů. Například při správné sazbě písmen „fi“ musí být „i“ velmi blízko k „f“. Problém je, že tečka nad „i“ koliduje s vrcholem „f“, a horní patka „i“ koliduje se střední linkou „f“. Proto se v písmech používají ligatury – slitky. Ligatura „fi“ je jediný znak, kterým se nahrazuje dvojice znaků „f“ a „i“. Většina písem obsahuje slitky fi a fl. Speciální písma obsahují často i další slitky, například ffl, ffi a beztečkové i.

Kapitálky

Kapitálky jsou písma, v nichž je velikost velkých písmen zmenšena na velikost malých. Používají se často k psaní výrazných nadpisů (a často je používá LaTeX). Při psaní nadpisu pomocí kapitá-lek se typicky první písmeno napíše normálně velké, zbytek pak kapitálkami. Výhodou oproti použití všech písmen velkých je mnohem lepší čitelnost (psaní jen velkými písmeny je nejen v typografii veliký prohřešek).

Speciální písma

Speciální písma obsahují různá vylepšení – slitky, ornamenty, kapitálky a ozdobné kapitálky (okrasná, kaligrafická písmena).

Metriky a tvary písem

Metriky písem definují rozestupy mezi znaky u písem s proměnnou šířkou. Obsahují informace

o velikosti písma a informace o párování písmen (tzv. kerning), které definují dvojice písmen, jež budou používat speciální rozestup. Například písmena „To“ typicky představují stlačovaný pár, protože při správném rozestupu je „o“ poněkud podsazeno pod „T“. Tyto informace jsou potřeb-né pro sázecí programy jako LaTeX kvůli zalamování řádků a stránek. Podobně jsou nutné pro publikační WYSIWYG programy. Další důležitou komponentou písma jsou obrysy nebo tvary. Jed-notlivé znaky v písmech se označují jako glyphy.

Technologie písem

V této kapitole naleznete jak neužitečné (v současnosti), tak užitečné informace o vývoji techno-logie písem, základní charakteristiky vybraných typů a dynamiku trhu, která napomohla výběru těch nejpoužívanějších. V současnosti se pravděpodobně již nesetkáte s písmy Type 1, Type 3 a Type 42. Shrnutí je, že de-facto standardem jsou písma True Type. Linux obsahuje podporu této technologie v knihovně FreeType. Někdy můžete nalézt některá rastrová písma pro použití na obrazovce, ale nikdy ne pro tisk.

Rastrová písma

Rastr (bitmapa) je matice bodů. Rastrová písma jsou reprezentována přesně tímto způsobem – jako matice bodů. Proto jsou závislá na zařízení – správně fungují jen v určitém rozlišení. Písmo urče-né pro obrazovku s rozlišením 75 DPI bude mít stále 75 DPI i na tiskárně s rozlišením 1 200 DPI.

Existují dva typy rastrových písem – tisková písma, například písma pk generovaná programem dvips, a obrazovková písma, používaná v systému X Window a na konzole. Soubory s obrazov-kovými písmy mají obvykle přípony bdf.gz nebo pcf.gz. Obrazovková rastrová písma mají nej-větší význam pro terminálová okna, konzoly a textové editory, kde nevadí nemožnost změnit jejich velikost a nemožnost je rozumně vytisknout.

Písma TrueType

Písma TrueType vyvinula společnost Apple. Svůj formát zpřístupnila společnosti Microsoft a úspěš-ně tak zaútočila na nadvládu společnosti Adobe na trhu s písmy. Písma TrueType ukládají metriky i definice tvarů v jediném souboru (typicky s příponou ttf). Nedávno byly vyvinuty font servery, které umožňují v systému X Window práci s písmy TrueType. Již delší čas tato písma podporuje formát PostScript i program ghostscript. Díky tomu popularita písem TrueType v Linuxu roste.

Písma Type 1

Standard Type 1 byl navržen společností Adobe a podporuje jej standard PostScript společnosti Adobe. Díky tomu jsou tato písma dobře podporována i v Linuxu. Podporují je systémy X Win-dow a ghostscript. Postscriptová písma tradičně představují v UNIXech správnou volbu ohledně čehokoliv, co zahrnuje tisk. Písma Type 1 se obvykle v UNIXu distribuují jako soubory afm (adobe font metric) a obrysové (outline) soubory, obvykle pfb (printer font binary) nebo pfa (printer font ascii). Obrysový soubor obsahuje definice tvarů, metrikové soubory definují metriky.

Písma Type 1 pro jiné platformy mohou být distribuována v jiných formátech. Například postscriptová písma pro Windows často používají pro definice metrik odlišný souborový formát pfm.

Písma Type 3

Tato písma se distribuují obdobně jako písma Type 1 – jako dvojice afm souboru s metrikami a pfa souboru. I když jsou podporována standardem PostScript, nepodporuje je systém X Win-dow, a jejich použití je tedy omezené.

Písma Type 42

Písma Type 42 jsou ve skutečnosti normální písma TrueType s hlavičkami, které umožňují jejich vykreslení interpretem PostScriptu. Řada aplikací (například ghostscript) s těmito písmy transpa-rentně pracuje. Pokud ale máte postscriptovou tiskárnu, budete možná muset tato písma explicit-ně vytvořit.

Type 1 versus TrueType – srovnání

Navzdory historickým sporům mezi producenty písem Type 1 a TrueType mají oba formáty hodně společného. Obojí jsou škálovatelná obrysová písma. Písma Type 1 definují tvary znaků pomocí kubických křivek, na rozdíl od písem TrueType, které používají kvadratické křivky. Teoreticky je to výhoda, protože křivky písem Type 1 tak představují nadmnožinu křivek písem TrueType. V praxi je rozdíl zanedbatelný.

Písma TrueType mají jednoznačnou výhodu v lepší podpoře hintingu (i písma Type 1 nabízejí možnost hintingu, ale není tak dobrá jako u písem TrueType). Tato funkce má ale význam pouze u zařízení s malým rozlišením, jako jsou obrazovky (lepší

hinting není už při rozlišení 600 DPI patrný, a to ani při malé velikosti písma). Dalším bodem, díky němuž je tato zjevná výhoda poně-kud problematická, je to, že dobře hintovaná písma TrueType jsou velice vzácná. Softwarové balí-ky s podporou hintingu jsou totiž pro většinu „malonávrhářů“ nesmírně drahé. Jejich použití si tedy mohou dovolit pouze společnosti, jako je například Monotype.

Hlavní rozdíl mezi písmy TrueType a Type 1 tak spočívá v jejich dostupnosti a podpoře aplika-cemi. Široká dostupnost písem TrueType pro Windows vedla k tomu, že řada webových stránek je navrhována s předpokladem, že konkrétní písma existují. Řada uživatelů Windows má navíc i mnoho dalších písem TrueType, protože jsou součástí různých aplikací. V Linuxu naopak větši-na aplikací podporuje písma Type 1 a úroveň podpory písem TrueType není tak vysoká. Navíc řada dodavatelů písem stále poskytuje písma ve formátu Type 1. Například Adobe nabízí jen velmi málo písem TrueType. Doporučuji používat to, co pro daný účel vyhovuje, a snažit se vyhnout konverzi z jednoho formátu do druhého (protože konverze vždy vede ke ztrátě kvality).

Kde získat písma pro Linux

TrueType Komerční programy

Písma TrueType se dají snadno získat, celou řadu typicky obsahují programy jako Microsoft Word nebo Word Perfect. Koupě programu Word Perfect je jednoduchá metoda, jak získat spoustu písem. (A pokud chcete ušetřit, stačí koupit nějakou starou verzi Word Perfectu pro Windows. Pře-číst písma z instalačního CD není žádný problém.)

Lucovy stránky

Stránky Luca Devroye (*http://cgm.cs.mcgill.ca/~luc/originalfonts.html*) obsahují odkazy na různé servery se zdarma distribuovanými písmy. Důležité je, že jde skutečně o zdarma distribuovaná písma, nejde o „warez“.

Webové stránky s písmy

Existuje několik stránek, které nabízejí písma ke stažení zdarma. Odkazy na řadu archivů nalez-nete například na adrese *http://www.freewareconnection.com/fonts.html.*

Výrobci

Řada výrobců prodává písma TrueType. Většina z nich je ale poměrně drahá a za stejné peníze rozumněji pořídíte písma Type 1. Více v kapitole věnované písmům Type 1. Jedna možnost, jak koupit levná písma TrueType, je adresa *http://www.buyfonts.com/.* Před koupí levných písem si přečtěte kapitolku o etice.

Písma Type 1 a Metafont Práce s formáty Mac a Windows

Řada výrobců dodává písma pro uživatele Windows a Maců. Občas tak může docházet k problé-mům. Typicky se s „písmy pro Windows“ pracuje velmi snadno, protože jsou zabaleny jako sou-bor .zip. Jediné, co je nutné udělat, je konverze souboru pfm na afm (pomocí pfm2afm).

Písma pro Macintosh jsou problematičtější, protože jsou typicky ve formátu .sit.bin – tedy v archivu stuffit. Bohužel, pro Linux neexistuje žádný nástroj, který by uměl rozbalit novější verze archivu stuffit. Jedinou možností je spustit Executor (emulátor Macu) nebo zkusit stuffit v dosemu nebo ve Wine. Po rozbalení souboru sit.bin je možné písma zkonvertovat nástrojem t1unmac, který je součástí balíčku t1utils.

Někteří dodavatelé bohužel nabízejí písma pouze ve formátu Macintosh (archivy stuffit). Podle Luca Devroye naštěstí většina z nich nabízí písma Type 1 i ve formátu pro Windows.

Zdarma dostupné

Ctan (*http://www.ctan.org/*) nabízí řadu dobrých písem, mnoho z nich je zdarma. Většina je ve for-mátu Metafont, některé jsou ale Type 1. Bluesky (*http://www.bluesky.com/*) nabízí zdarma Type 1 verzi písem Computer Modern. (Jejich kvalita je vynikající, cokoliv ve srovnatelné kvalitě a úpl-nosti by stálo kolem 500 dolarů.)

Stránky Luca Devroye (*http://cgm.cs.mcgill.ca/~luc/originalfonts.html*) obsahují odkazy na různé servery se zdarma distribuovanými písmy. Důležité je, že jde skutečně o zdarma distribuovaná písma, nejde o „warez“.

URW uvolnil standardní postscriptová písma používaná ve většině tiskáren. Jde o docela kvalitní písma. Další zdarma dostupná a sharewarová písma nabízí Walnut Creek Archive (ftp://ftp.cdrom.com/ pub/os2/fonts/). Některá z nich jsou zjevně vykradená (a v nevysoké kvalitě). Pokud součástí písma není i nějaká distribuční licence, jde obvykle o vykradené písmo. Různá písma Type 1 nabí-zí i Winsite (*http://www.winsite.com/win3/fonts/atm/*). Bohužel, u některých písem jsou chyby v souborech afm a chybí kerning. (Soubory afm je možné opravit editací části FontName v sou-boru, která musí odpovídat názvu písma v definičním souboru písma. Vytvoření kerningových dvojic je pak docela mimo rámec tohoto dokumentu.)

Na stránkách Luca Devroye (*http://cgm.cs.mcgill.ca/~luc/*) najdete několik písem, která sám navr-hl, dále řadu odkazů a nesmírně zajímavé informace na typografická témata. Jde o stránky, které byste rozhodně neměli minout.

Komerční písma

Cena versus kvalita: Proč kupovat drahá písma?

Ptáte se, proč jsou některá písma hrozně drahá a jiná levná? Jde většinou o „standardní postscrip-tová písma“, která jsou součástí většiny postscriptových tiskáren. Další známá otázka – proč kupo-vat dražší písma? Myslím si, že pro příležitostného uživatele stačí levnější písma, která bývají často součástí různých CD. Pokud ale používáte písma pro „opravdovou práci“ nebo jste prostě blázen do typografie, pak se bez vysoce kvalitních písem neobejdete. Některá velmi kvalitní písma jsou zdarma (například Computer Modern), jiná pak stojí obrovské peníze.

Výhoda levnějších písem je zjevná – jsou levná. Kvalitní písma mají ale také své výhody.

- Etické hledisko: Levnější písma jsou často vykradená. Návrh kvalitního písma vyžaduje dlouhou dobu a zkušeného návrháře. Písma prodávaná pod jeden dolar nejsou téměř jistě originální. CD se spoustou různých písem jsou téměř vždy různě vykradená (výjimkou jsou CD některých výrobců, která ale stojí pár stovek dolarů). Vykradená písma mají typicky výrazně nižší kvalitu než originály.

Úplnost: Velmi kvalitní písma (například od Adobe) obsahují různé varianty s řadou pěk-ných doplňků, rozšiřujících rodinu písma. Často obsahují tučné a polotučné varianty, kur-zivu, různé typy kapitálek a slitků.
Kvalita: Řada zdarma šířených písem a různě vykradených písem postrádá základní funkce jako párování (kerning) nebo slitky. Jde typicky o levné kopie. Naproti tomu renomovaní návrháři pečlivě studují původní návrh a zpracovávají jej podle svých nejlepších možností.
Autenticita: Autor písma Adobe Garamond (konkrétně Robert Slimbach) pečlivě studoval původní návrhy Clauda Garamonda. Renomovaní návrháři vždy stavějí své návrhy na pečli-vém průzkumu, nestačí jim pouze něco stáhnout na Internetu a upravit to Fontographerem.

Cena

Dobrý zdroj CD s různými písmy Type 1 v rozumné kvalitě je Bitstream (*http://www. bitstream.com/*). Nejznámějšími produkty jsou 250 font CD (*http://www.bitstream.com/ products/world/font_cd/bits_collection.html*) a 500 font CD (*http://www.bitstream.com/ products/world/font_cd/500_cd.html*). To druhé stálo v době vzniku tohoto textu 50 dola-rů. Většina písem v produktech Corel je licencována právě od Bitstreamu.
Matchfonts (*http://www.matchfonts.com/*) nabízí poněkud dražší písma, distribuovaná v „balících“ po osmi za 30 dolarů. Nabízejí některá pěkná kaligrafická písma. Všechna jsou dodávána ve snadno použitelném formátu (písma ATM pro Windows se dodávají v .exe souborech, ale nenechejte se příponou zmást – jde o běžné archivy zip).
EFF (*http://www.buyfonts.com/*) prodává písma TrueType po dvou dolarech. Nabízejí také „profesionální“ fonty PostScript aTrueType po 16 dolarech.

Kvalita

Adobe nabízí několik velmi kvalitních písem na *http://www.adobe.com/type/*. Některá jsou velmi drahá, ale nabízejí i cenově dostupnější – viz *http://www.adobe.com/type/collecti-ons.html*. Adobe nabízí jedny z nejúplnějších rodin písem na trhu, například Garamond (*http://www.adobe.com/type/browser/P/P_912.html*), Caslon (*http://www.adobe.com/* ty-pe/browser/P/P_180.html) a jejich různé mutace.
Berthold Types Limited (*http://www.bertholdtypes.com/*) nabízejí různá kvalitní písma. Něk-terá z nich prodává i Adobe, za stejných cenových podmínek je však nabízí přímo Bert-hold.
ITC (*http://www.itcfonts.com/*) nabízí několik kvalitních písem (některá z nich nabízí ve svých produktech i Corel). Celé rodiny písem stojí kolem 100–180 dolarů. Dodávají se ve formátech Type 1 i TrueType, rozumnější je zvolit distribuci pro Windows, protože formát pro Macy se v Linuxu zpracovává hůře.
Známým dodavatelem písem je Linotype (*http://www.linotypelibrary.com/*), nabízí písma legendárních návrhářů, jako byl Herman Zapf (to je ten, po němž se jmenuje písmo „Zapf Chancery“, navrhl například i Palatino).
Monotype (*http://www.monotype.com/*) je autorem většiny písem používaných v produk-tech Microsoft.
Tiro Typeworks (*http://www.portal.ca/~tiro/*) prodávají kvalitní, i když poněkud drahá písma. Jejich písma jsou velmi úplná, obsahují slitky, kapitálky, nadpisová písma a podob-

ně. Nabízejí dokonce formát přímo pro UNIX – což je příjemné překvapení poté, co všude vidíme volbu „Windows nebo Mac“.

Další odkazy

Odkazy na spoustu dalších stránek naleznete na stránkách Luca Devroye, *http://cgm.cs.mcgill.ca/ ~luc/*.

Užitečný linuxový software pro práci s písmy

Pro práci s písmy v Linuxu existuje řada programů. Některé z nich se však již nepoužívají.

chkfontpath je nástroj pro práci s konfiguračními soubory xfs.
fontinst, *http://www.tug.org/applications/fontinst/index.html*, je nástroj usnadňující instala-ci fontů Type 1 do LaTeXu.
freetype, *http://www.freetype.org/*, je TrueType knihovna dodávaná s většinou distribucí Linuxu.
Ghostscript, *http://www.cs.wisc.edu/~ghost/*, je program pro tisk v Linuxu. Verze dodávaná s Linuxem je GNU ghostscript. Jde o starší verzi Alladin ghostscript (jejichž předchozí verze byly licencovány pod GPL).
pfm2afm, *http://pegasus.rutgers.edu/~elflord/font_howto/pfm2afm.tgz*, je nástroj pro kon-verzi metrikových souborů pfm používaných ve Windows na soubory afm používané v Linuxu. Je založen na původní verzi z archivu CTAN s úpravami Roda Smithe, které umožňují překlad nástroje pod Linuxem.
mminstance a *t1utils*, *http://www.lcdf.org/~eddietwo/type/*, jsou dva balíky pro práci s písmy Type 1. Balík mminstance slouží k práci s písmy multiple master od Adobe, t1utils je sada nástrojů pro konverzi mezi různými formáty písem Type 1.
ttf2pt1, *http://quadrant.netspace.net.au/ttf2pt1/*, je nástroj pro konverzi písem TrueType na Type 1. Je užitečný v případě, kdy používáte aplikace, které pracují pouze s písmy Type 1.
ttfps, ftp://ftp.dcs.ed.ac.uk/pub/jek/programs/ttfps.tar.gz, konvertuje písma TrueType na Type 42.
ttfutils, *http://pegasus.rutgers.edu/~elflord/font_howto/ttfutils-0.2.tar.gz*, je balík nástrojů pro práci s písmy TrueType. Vyžaduje přítomnost programu ttf2pt1. Velmi užitečný, ne-li nezbytný.
type1inst, ftp://ftp.metalab.unc.edu/pub/Linux/X11/xutils/, základní balík pro instalaci pí-sem Type 1. Významně usnadňuje instalaci.
xfstt, ftp://ftp.metalab.unc.edu/pub/Linux/X11/fonts/, font server s podporou TrueType. Velmi šikovný, lepší volba je ale xfs.
xfsft, *http://www.dcs.ed.ac.uk/home/jec/programs/xfsft/*, font server, je součástí xfs.
x-tt, *http://hawk.ise.chuo-u.ac.jp/student/person/tshiozak/x-tt/*, font server navržený pro práci s korejskými a japonskými fonty.

Etické a licenční problémy

Licencování písem je velmi sporný problém. Je pravda, že existuje spousta zdarma šířených písem, avšak pokud neobsahují vlastní licenční podmínky, je velmi vysoká pravděpodobnost, že se jedná o písma nějakým způsobem „vykradená“. Celý problém se komplikuje díky právní úpravě ohledně intelektuálního vlastnictví ve vztahu k písmům. V USA v zásadě platí, že soubory s písmy jsou chráněny autorským právem, zatímco samotný vzhled písma nikoliv. Jinak řečeno – je nelegální písmo šířit v binární podobě, avšak je naprosto legální provést „reverzní překlad“ písma jeho vytiš-těním na papír a návrhem křivek, které budou výtisku odpovídat. Takto vytvořená písma jsou typicky levná nebo zdarma, jejich kvalita ale bývá nízká. Taková písma bývají společně s pirát-skými verzemi písem distribuována na laciných CD. Nelze jednoduše poznat, zda se jedná o pirát-ské písmo nebo o „reverzní překlad“. Díky této situaci je nesmírně komplikované jakékoliv (legál-ní) šíření domněle zdarma distribuovaných písem.

Jednou z nejnepříjemnějších věcí na tomto písmovém pirátství je, že se tím znehodnocuje práce návrhářů originálních písem. Pirátská písma jsou typicky distribuována na CD po tisících, bez zmínky o původních autorech. Naproti tomu u legálně šířených písem bývají uvedeni původní návrháři.

Na tento problém existuje celá řada rozdílných názorů. Na stránkách *http://www.typeright.org/* naleznete vysvětlení problematiky intelektuálního vlastnictví. Opačný názor prezentuje Southern Software, *http://www.ssifonts.com/* – jejich písma si ale nekupujte. Písma Type 1 nabízené touto společností (nekvalitní kopie písem Adobe) neobsahují AFM, a jsou tedy nepoužitelné.

Další názory na téma intelektuálního vlastnictví ve vztahu k písmům najdete na *http://www. faqs.org/faqs/fonts-faq/part2/* a *http://cgm.cs.mcgill.ca/~luc/*. Tyto odkazy představují méně extrémní skupinu názorů.

Odkazy

Informace o písmech

Stránky Luca Devroye, *http://cgm.cs.mcgill.ca/~luc/*, obsahují tolik informací o písmech a dalších věcech, že by to potopilo loď. Autor stránek navrhl řadu zdarma distribuovaných písem a nabízí spoustu zajímavých odkazů, informací a komentářů.
Seznam vysoce kvalitních písem Scribus, *http://www.scribus.org.uk/modules.php?op =modload&name=Web_Links&file=index&req=viewlink&cid=3*, je Open Source projekt pro DTP.
Stránka Jima Landa, *http://www.geocities.com/SiliconValley/5682/postscript.html*, obsahuje odkazy na informace o postscriptu a písmech.
Spoustu otázek ohledně písem řeší *http://www.faqs.org/faqs/fonts-faq/*.
The (preliminary) TrueType HOWTO, *http://www.moisty.org/~brion/linux/TrueType-HOWTO.html*, neúplný dokument z června 1998. Uvádíme jej zde pouze pro úplnost.

Informace o postscriptu a tisku

Standard PostScript popisuje *http://www.adobe.com/print/postscript/main.html.*
Domovská stránka Ghostscriptu, *http://www.cs.wisc.edu/~ghost/*, obsahuje řadu informací a nejnovější tiskové ovladače.

Stránky Jima Landa, *http://www.geocities.com/SiliconValley/5682/postscript.html*, obsahují řadu odkazů na stránky věnované PostScriptu a písmům.
Printing FAQ Christophera Brownea, *http://www.hex.net/~cbbrowne/printing.html.*

Slovníček

afm

Adobe Font Metric. Tento soubor obsahuje informace o šířkách a mezerách vztahujících se k písmu. Nedefinuje tvary znaků.

anti-aliasing

Technika pro vykreslování písem na zařízeních s malým rozlišením (např. monitorech). Problém při vykreslování spočívá v tom, že znak je tvořen křivkami, ty se ale vykreslují po bodech. Jed-noduché řešení spočívá v tom, vykreslit černě všechny body uvnitř znaku, ostatní nechat bílé. Tím se ale neřeší problém bodů na hranici znaku. Chytřejší algoritmy vykreslují hraniční body různými odstíny šedi – a to je právě antialiasing neboli vyhlazování hran.

bdf fonts

Rastrová písma používaná v X Window.

bezpatkové písmo

Písma bez patek (sans je francouzsky bez). Tato písma mají výraznější (sans-serif) vzhled a použí-vají se k sazbě titulků. I když učebnice typografie udávají jejich použití jen pro titulky, dají se pou-žít i jinak. Některá bezpatková písma jsou navržena s ohledem na čitelnost a ne na výraznost. Pou-žívají se například k sazbě katalogů nebo marketingových materiálů. Písmo Verdana používá Mic-rosoft kvůli dobré čitelnosti při malé velikosti na zařízeních s nízkým rozlišením. Mezi známá bez-patková písma patří Lucida Sans, MS Comic Sans, Avant Garde, Arial, Verdana a Century Gothic.

bitmap fonts viz rastrová písma didone viz modern DPI (Dots Per Inch), bodů na palec. Jednotka rozlišovací schopnosti zařízení. Monitory typicky zobrazují 75–100 DPI, moderní tiskár-

ny 300–1 200 DPI.

expert fonts

Kolekce doplňujících znaků, které písmo rozšiřují. Zahrnují obvykle kapitálky, ornamenty, speciální ligatury a číslice s různou šířkou. Jsou součástí většiny písem Adobe.

font server

Program, který zpřístupňuje písma X serveru, například X.org nebo Xfree86.

glyph

Zajímavé slovo označující tvar. Jedná se o komponenty, z nichž je tvořen obrys znaku. Například tečka nad „i“ je jeden glyph, dalším je svislá čára, jiným zase patky.

charset, znaková sada

Skupina 8bitových glypthů. Znaková sada ISO-8859-1 (Latin 1) obsahuje znaky pro západoevropské jazyky, ISO-8859-2 obsahuje hebrejské znaky, ISO-8859-5 obsahuje znaky cyrilice atd. Tento způsob používání je již zastaralý díky rozvoji kódování Unicode. Základní knihovna jazyka C (libc) obsahuje nástroje pro konverzi textu z jedné znakové sady do jiné a do a z kódování Unicode.

ISO-8859

Standard ISO-8859 zahrnuje některá 8bitová rozšíření k původní znakové sadě ASCII (rozšíření se také nazývají ISO 646-IRV). Existuje více variant, jako například ISO-8859-1 (neboli Latin 1), ISO-8859-2 (Latin 2) atd. Tento standard se v současnosti využívá hlavně na platformě Windows, nic-méně i zde je nahrazován univerzálnějším a úplnějším standardem Unicode, zvláště pak znako-vou sadou UTF-8. Při použití znakové sady ISO-8859 není možné v jediném dokumentu míchat více jazyků, například hebrejštinu s portugalštinou, arabštinu s francouzštinou, chorvatštinu s něja-kým severským jazykem, japonštinu s angličtinou atd. Více informací naleznete na stránkách man Linuxu.
ISO-8859-1 neboli Latin 1 Standard ISO-8859-1 (nebo jednodušeji Latin 1) je znaková sada, která definuje speciální znaky s ordinálním číslem větším než 128, které se používají v západoevropských zemích u jazyků, jako je portugalština, španělština nebo francouzština. Prvních 128 znaků jsou standardní znaky ASCII. Tato znaková sada obsahuje znaky jako „ç“, „á“, „É“, „ü“, „î“, ale nezahrnuje znak euro, „€“, který

je až součástí aktualizace ISO-8859-15. Tato znaková sada je již zastaralá a v textu, webových strán-kách nebo složitých dokumentech se doporučuje použít znakovou sadu UTF-8. kerning viz párování ligatura viz slitek medievalová písma

Tradiční skupina písem. Jsou založena na návrzích až z 15. století. Jsou vynikající pro sazbu dlou-hých textů, např. knih. I když jejich návrh vychází z velmi starých tradic, některá z nich vznikla poměrně nedávno. Například písmo Goudy Old Style navrhl

Goudy počátkem 20. století. Klasic-kými představiteli jsou Goudy Old Style, Garamond a Caslon.

metafont

Grafický jazyk používaný k definici písem. Metafont má řadu sympatických funkcí, hlavní z nich je ta, že změna velikosti písem nemusí být lineární. Znamená to, že písmo o velikosti 17 bodů není prostou zvětšeninou písma o velikosti 10 bodů. Před příchodem technologie multiple master od Adobe byla tato funkce jedinečná právě pro metafont. Výhodou tohoto jazyka je vytváření velmi kvalitních písem, nevýhodou je pomalé generování rastrů, takže nejsou vhodné pro WYSIWYG publikování.

metrika

Obsahuje informace o mezerách mezi znaky. Metrika je něco jako rámeček, do nějž lze znak vykreslit. Metrika je nezbytná pro potřeby sazby znaků na stránce, definice tvarů sama o sobě k sazbě není zapotřebí. Písma s proměnnou šířkou proto obsahují jak definice tvarů, tak definice metrik. Metrika navíc obsahuje informace o kerningu.

modern fonts

Písma založená na návrzích z 19. století a novější. Mají výrazný vzhled díky svislému zvýraznění. Mají výraznější charakter a vzhled než písma medievalová a tranzitivní, stále si však zachovávají jistou formálnost. Nejsou vhodná pro delší dokumenty, používají se pro výraznější kratší texty. Typickým písmem této kategorie je Bodoni.

old style fonts viz medievalová písma patkové písmo

Písmo s krátkými čárkami (patkami) na koncích znaků, které typicky zvyšují čitelnost písma. Tento typ písma se velmi obtížně zobrazuje na zařízeních s malou rozlišovací schopností, zejména při malé velikosti písma. Proto bývají na těchto zařízeních mnohdy čitelnější bezpatková písma. Některá patková písma (tzv. moderní) nejsou vhodná pro sazbu dlouhých dokumentů.
pcf

Rastrová písma používaná v systému X Window.

písmo s plochou patkou

Též slabserif nebo egyptienka. Skupina písem, jejichž patky vypadají jako bloky. Jsou obvykle, ale ne vždy, velmi dobře čitelná. Působí výrazným vzhledem. Známými příklady jsou písma Claren-don, New Century Schoolbook a Memphis.

PostScript

Programovací jazyk určený k tvorbě stránek. Jde o ochrannou známku autora, společnosti Adobe, ale zároveň o standard ISO. Ke zobrazení PostScriptu je nutný interpret. Tím může být počítačový program, například Ghostscript, nebo to přímo umějí některé tiskárny.

rastrová písma

Tato písma jsou jednoduše sbírka bodů. Každý znak je reprezentován maticí bodů. Díky tomu jsou rastrová písma závislá na rozlišení zobrazovacího zařízení, a totéž rastrové písmo tedy nejde pou-žít na obrazovce i na tiskárně. Příkladem rastrových obrazovkových písem jsou písma pcf a bdf v systému X Window. Tisková rastrová písma jsou například písma PK v TeXu.

sans serif viz bezpatkové písmo serif viz patkové písmo slab serif viz písmo s plochou patkou slitek

Speciální znak sloužící ke zobrazení dvojice znaků. Nejlépe se vysvětluje na příkladu: Při vykreslení dvojice znaků „fi“ tečka nad „i“ koliduje s vrcholem „f“ a horní patka „i“ koliduje se střední linkou „f“. Slitek fi je jediný znak používaný jako náhražka dvojice za sebou následujících znaků „f“ a „i“. Dalšími slitky jsou třeba fl ffi a ffl. Většina písem obsahuje pouze slitky fi a fl. Další slitky bývají součástí expertních písem.
párování též kerning U písem s proměnnou šířkou jsou mezi různými znaky různé mezery. Metrika písma definuje mezery mezi různými dvojicemi znaků, takzvané kerningové páry.
tranzitivní písma

Písma s modernější podobou než klasická medievalová písma. Většina z nich je velmi dobře čitel-ná. Příkladem jsou písma Baskerville a Times Roman.

Type 1

Typ písem navržený společností Adobe. Je podporován většinou linuxových aplikací, protože jde
o typ již dlouho podporovaný standardem PostScript a X servery. Distribuují se v řadě různých formátů. Na Unixech typicky jako afm soubor s metrikou a pfb (printer font binary) nebo pfa (printer font ascii) soubor definující tvary.

Type 3

Formát podobný formátu Type 1. Přípony jsou obdobné jako u Type 1, není však podporován v X Window, a proto s ním pracuje jen velmi málo linuxových aplikací.

Unicode

Před příchodem kódování Unicode byl každý znak reprezentovaný jedním bajtem, což umožnilo mít k dispozici 256 znaků. Znak

s ordinálním číslem 0xE2 v hexadecimální soustavě ve znakové sadě Latin-1 reprezentuje „â“, znak se stejným ordinálním číslem ve znakové sadě ISO-8859-7 (řecká znaková sada) reprezentuje znak „â“. Unicode využívá vícebajtové znaky s cílem, aby každý znak v každé civilizaci a v každém jazyku byl mapován na jedinečný vícebajtový hexadecimální kód. V našem příkladu má znak „â“ ordinální číslo 0x00E2 a znak „â“ 0x03B2.

UTF-8

UTF-8 je kódování znakové sady Unicode, které využívá pouze jeden bajt pro znaky ASCII, dva bajty pro znaky znakové sady Latin-1 (ISO-8859-1) s ordinálními čísly vyššími než 128 a tři nebo čtyři bajty pro ostatní znaky. Soubor, který obsahuje anglický text v kódování UTF-8, je bajtově identický s jeho verzemi Latin-1 a ASCII. Pokud jsou ve stejném souboru použity jiné znaky, každý takový znak bude vícebajtový, s prefixem escape bajtů v kódování UTF-8. Moderní aplikace jako OpenOffice.org vytvářejí dokumenty v kódování UTF-8. Kódování UTF-8 je vhodné používat při vytváření souborů textů nebo HTML. Moderní linuxové instalace využívají kódování UTF-8 v pro-středí pro jakoukoliv zemi a jakýkoliv jazyk a je v současnosti de-facto standardem pro reprezen-taci textu. Správce systému musí mít velmi dobrý důvod, proč kódování UTF-8 nepoužít.

Další informace

Překlad zdrojů FreeType pro BCI

„Hinting“ je specifická vlastnost písem TrueType, která zlepšuje jejich vzhled. Naneštěstí jsou s jejím použitím spjaty jisté licenční a patentové problémy a ve zdrojových souborech freetype je ve výchozím stavu podpora těchto funkcí deaktivovaná. Stejně tak tomu je i v případě binárních souborů od externích dodavatelů. Pro aktivaci těchto funkcí je nutné provést nový překlad zdro-jů FreeType. Používáte-li balíčky z externích (neoficiálních) distribučních repozitářů, pravděpo-dobně již budou mít tuto vlastnost aktivovánu.

Na jakémkoliv systému

Ve zdrojových souborech vyhledejte soubor include/freetype/config/ftoption.h. V tomto souboru vyhledejte text:

```
/* #define TT_CONFIG_OPTION_BYTECODE_INTERPRETER */
```

a jednoduše jej odkomentujte tak, aby řádek vypadal takto:

```
#define TT_CONFIG_OPTION_BYTECODE_INTERPRETER
```

Na systémech Red Hat (včetně systémů Fedora)

Uživatelé systému Red Hat mohou provést překlad balíčku RPM se zdroji FreeType úpravou nastavení (ostatní distribuce, které využívají balíčky RPM, používají stejný způsob):

Text:

```
%define without_bytecode_interpreter        1
```

je nutné upravit tatko:

```
%define without_bytecode_interpreter        0
```

Ostatní dodavatelé využívají podobný, stejně jednoduchý mechanismus.
Pak stačí provést překlad a instalaci nově vytvořených binárních souborů. Po instalaci je nutné provést restart X serveru, protože X server má kód FreeType nahrán do paměti.

Překlad balíčku RPM pro vaši distribuci

Pokud je vaše distribuce uvedena v seznamu distribucí, ale pro vaši platformu nemůžete nalézt binární podobu balíčku (například pro platformu x86_64), můžete si vytvořit kompatibilní balíček RPM vlastními silami. Je nutné provést následující kroky:

Mít k dispozici překladač a vývojové balíčky.
Stáhnout zdrojový soubor RPM (přípona .src.rpm) pro vaši distribuci.
Pod účtem uživatele root provést následující příkaz:

```
bash# rpmbuild —rebuild [právě stažený soubor s příponou .src.rpm]
```

Nalézt binární balíček RPM pro vaši platformu ve složce /usr/src/rpm nebo /usr/src/redhat.
Poslat nám je, abychom je mohli uveřejnit v této dokumentaci.

Potřebujeme vaši pomoc

Ano, potřebujeme i vaši pomoc, takže prosím zašlete zprávu na adresu avi@unix.sh s informace-mi o tom, s čím chcete přispět. V současnosti je nutné:

Vytvořit balíčky FreeType s aktivovaným interpretrem BCI pro různé verze linuxových distribucí (tak jak to my děláme pro Fedoru).
Poskytovat kvalitní balíčky Webcore Fonts pro různé verze distribucí.
Upřesnit Tabulku 1.
Vytvořit návod na překlad balíčků i pro jiné typy balíčků než RPM (například pro balíčky .debs nebo Slackware).
Přeložit tento dokument do jiných jazyků.
Rozšířit tento dokument.

Samba – brána s autentizačními funkcemi

Úvod

Tento návod si klade za cíl objasnit proces vytvoření linuxové brány nebo firewallu, který při při hlášení nebo odhlášení uživatelů z pracovních stanic se systémem Windows upravuje pravidla. V tomto návodu naleznete informace o tom, jak vytvořit bránu tak, aby prováděla pro pracovní stanice se systémem Windows funkci NAT (překlad síťových adres) nebo MASQUERADE (skrytí adres vnitřní sítě za jednu veřejnou adresu). Tuto konfiguraci je samozřejmě možné přizpůsobit pro libovolnou konfiguraci sítě a také pro povolení nebo zakázání přístupu ke službám, serverům nebo celým podsítím.

Představte si, že máte za úkol vytvořit bránu, která by umožnila pracovním stanicím s operačním systémem Windows přístup k Internetu a umožnila také autentizaci uživatelů předtím, než uživa-telé začnou přistupovat do externích sítí. První řešení, které vás určitě napadne, je použít Squid. To je určitě výborné řešení v případě, kdy uživatelé vystačí s přístupem přes http a ftp. Pokud budou přistupovat i k jiným službám, jako je např. pop, smtp, ssh, nebo k nějakému databázo-vému serveru, využije se NAT nebo MASQUERADE. Co ale s autentizaci uživatelů?

Zde nabízím své řešení, které umožní autentizaci uživatelů a kontrolu nad přístupem uživatelů do externích sítí.

Základní informace

Víme, že Samba může fungovat jako řadič domény, takže umožní provádět autentizaci uživatelů na pracovních stanicích s operačním systémem Windows. Samba jako řadič domény umožní na pracovních stanicích provádět přihlašovací skripty. Tyto přihlašovací skripty je možné využít k při-pojení vybraného sdíleného disku z linuxového řadiče domény. Tento „vynucený“ sdílený disk může mít dále nakonfigurované skripty, které se provedou při přihlášení nebo odhlášení uživate-le. Existuje program s názvem smbstatus, který umí vypsat seznam používaných sdílených disků současně s uživatelským jménem a IP adresou pracovní stanice. Vše, co je nutné udělat, je vyhle-dat příslušné informace z výstupu programu smbstatus a odpovídajícím způsobem upravit pravi-dla firewallu.

Bonbónek

Pokud jste netrpěliví a neradi čtete, podívejte se na *http://sourceforge.net/projects/smbgate/*, ale pak se sem určitě vraťte.

Záruky a odpovědnost

Není možné přijmout jakoukoliv odpovědnost za obsah tohoto návodu. Koncepty, příklady a ostatní obsah tohoto návodu využíváte na vlastní nebezpečí. Protože toto je nové vydání návo-du, mohou se v něm vyskytnout chyby a nepřesnosti, které mohou mít destruktivní účinky na váš operační systém. Postupujte s opatrností, a i když je to vysoce nepravděpodobné, autor nenese za případné škody jakoukoliv odpovědnost.

Nové verze

Nejnovější verze tohoto návodu je k dispozici na adrese *http://ram.eti.br* nebo na adrese *http://www.tldp.org.* Odpovídající dokumenty HOWTO naleznete na domovské stránce Linux Documentation Project na adrese *http://tldp.org.*

Zpětná vazba

Připomínky a kritika jsou vítány. Pokud naleznete jakoukoliv chybu ve skriptech, dejte mi prosím vědět. Jsem k dispozici na adresách *ricardo@ram.eti.br* nebo *ricardo.mattar@bol.com.br.*

Znalosti a požadavky

Tento návod je určen pro příležitostné správce systémů. Pro pochopení obsahu je nutné mít znalosti z následujících oblastí:

TCP/IP,
nástroj netfilter,
skriptovací jazyk (bash?),
SAMBA a sítě a doménové řadiče v prostředí systému Windows.

K těmto tématům je naštěstí dost literatury na Internetu.

Použitý software

Na serveru je nutné mít nainstalovaný alespoň tento software:

Samba,
nástroj iptables,
skriptovací jazyk.

Literatura

Průvodce IPTABLES TUTORIAL od Oskara Andreassona, který naleznete na stránkách: *http://ip tables-tutorial.frozentux.net/.*

Dokumenty HOWTO přímo od týmu vývojářů Samby, které naleznete na stránkách:
http://us5.samba.org/ samba/docs/man/Samba-HOWTO-Collection/.

Konfigurace systému Linux

V tomto návodu se předpokládá, že máte jádro z řady 2.4 (a vyšší), protože využívá nástroje iptab-les. Kromě toho neexistuje jiné omezení, které by bránilo využívat systém v konfiguraci s jádrem verze 2.2 se skripty upravenými pro použití s nástrojem ipchains.

Samozřejmě je nutné instalovat uživatelské nástroje pro práci s iptables, webový server Apache (pokud chcete využívat CGI nástroj pro změnu hesel) a samozřejmě Sambu. Jádro musí být pře-loženo s podporou pro moduly iptables.

Možná budete chtít využít DHCP. Pokud ano, nastavení je velmi jednoduché. Je však nutné nakonfigurovat server DHCP tak, aby klientům poskytoval IP adresu názvového serveru a IP adresu výchozí brány. Počítače se systémem Windows tyto informace nutně potřebují.

Základní nastavení

Pro účely vytvoření brány s funkcemi autentizace vyhoví obecně jakákoliv základní konfigurace systému, nainstalovaného z některé linuxové distribuce. Je pouze nutné ověřit instalaci Samby a nástroje iptables.

Hierarchie složek

Pro dokončení konfigurace bude nutné vytvořit následující hierarchii složek: Tato složka se využije pro

sledování uživatelů a IP adres:

/var/run/smbgate/

Do této složky se budou ukládat uživatelské skripty:

/etc/smbgate/users/

Do této složky se budou ukládat skripty pro skupiny:

/etc/smbgate/groups/

Sdílená složka netlogon:

/home/samba/netlogon/

Složka pro sledování sdílení:

/home/samba/samba/

Skripty a démoni v tomto příkladu vyžadují výše uvedenou hierarchii složek.

Nastavení firewallu

Je velmi nepravděpodobné, že jádro vaší distribuce nebude přeloženo s podporou pro iptables a nebudou instalovány uživatelské nástroje. Pokud tomu tak není, na adresách *http://www. netfilter.org* a *http://www.iptables.org* naleznete jak vlastní software, tak dokumentaci.
Aby brána mohla správně pracovat, je nutné základní nastavení firewallu. Více informací nalez-nete v dokumentaci k nástroji iptables: (*http://iptables-tutorial.frozentux.net/iptables-tutorial.html*). Je to velmi zajímavé čtení. Zajímavé české materiály najdete například na adrese *http://www.petricek.cz/mpfw/.* Pokud nemáte čas na čtení dokumentace, můžete použít následují-cí kód, který obsahuje pouze základní pravidla, ale může vyhovět vašim potřebám:

```
#!/bin/sh IPTABLES=/usr/sbin/iptables /sbin/depmod -a /sbin/insmod ip_tables /sbin/insmod ip_conntrack /sbin/insmod ip_conntrack_ftp /sbin/insmod ip_conntrack_irc /sbin/insmod iptable_nat /sbin/insmod ip_nat_ftp echo “1” > /proc/sys/net/ipv4/ip_forward echo “1” > /proc/sys/net/ipv4/ip_dynaddr $IPTABLES -P INPUT ACCEPT $IPTABLES -F INPUT $IPTABLES -P OUTPUT ACCEPT $IPTABLES -F OUTPUT $IPTABLES -P FORWARD ACCEPT $IPTABLES -F FORWARD $IPTABLES -t nat -F
```

Určitě si všimnete, že kód nedělá nic jiného, než že nahraje moduly jádra, které zajišťují funkce nat a firewallu a zapne směrování paketů. Můžete (a rozhodně byste měli) doplnit další pravidla, kterými zajistíte standardní chování brány ve vašem síťovém prostředí, nicméně všechno ostatní zajistí skripty, volané démonem Samby.

Vezměte prosím na vědomí, že tento kód neobsahuje ani náznak zabezpečení. Zde uvedené příklady nepoužívejte v produkčním prostředí. Příklady jsou vytvářeny pouze pro výukové účely. Do skriptů je nutné doplnit dodatečnou konfiguraci firewallu, která bude v souladu s vaším konkrétním síťovým prostředím.

Právě jsem vás varoval!

Nastavení Samby

Zkontrolujte, jestli je nainstalovaná Samba. Pokud vaše distribuce Sambu neobsahuje ve výchozí konfiguraci, podívejte se na *http://www.samba.org*, kde naleznete instalační balíčky a dokumen-taci o instalaci Samby. Projděte si tyto webové stránky a sami si vyhledejte potřebné informace. Na webových stránkách Samby naleznete velké množství dokumentace a je možné, že dokumen-taci o Sambě obsahuje přímo vaše linuxová distribuce – balíček se obvykle jmenuje samba-doc.

Sambu je nutné nastavit tak, aby pracovala jako primární řadič domény. Příklad konfiguračního souboru naleznete v tomto návodu, ale přesto vám doporučuji, abyste si prošli dokumentaci *http://us4.samba.org/samba/docs/man/Samba-HOWTO-Collection/* a zjistili si všechny potřebné informace o primárním řadiči domény.

Základní nastavení Samby

Protože nechci přepisovat celou dokumentaci Samby, následuje vzorový soubor smb.conf s kon-figurací:

```
# Globální parametry [global] workgroup = DOMAIN netbios name = LINUX server string = Linux PDC encrypt passwords = Yes map to guest = Bad Password passwd program = /usr/bin/passwd unix password sync = Yes max log size = 50 time server = Yes socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192 add user script = /usr/sbin/useradd -d /dev/null -g 100 -s /bin/false -M %u logon script = %a.bat domain logons = Yes os level = 64 lm announce = True preferred master = True domain master = True dns proxy = No printing = lprng [homes] comment = Domovské složky path = /home/%u read only = No [printers] comment = Tiskárny path = /var/spool/samba printable = Yes browseable = No available = No [netlogon] comment = Sdílená složka NetLogon path = /home/samba/netlogon guest account = [samba] comment = Složka pro sledování sdílení path = /home/samba/samba browseable = No root preexec = /usr/local/bin/netlogon.sh %u %I root postexec = /usr/local/bin/netlogoff.sh %u
```

Další informace o konfiguraci serveru a sítě naleznete v dokumentaci Samby.

Přihlašovací skript

Pokud použijete nastavení "logon script = %a.bat", Samba zjistí aktuální operační systém počí-tače, z něhož se uživatel přihlašuje, a zavolá odpovídající přihlašovací skript. Pokud chcete pou-žít statický skript, změňte nastavení na "logon script = netlogon.bat". V této oblasti si může-te konfiguraci upravit dle libosti, a dokonce můžete přihlašovací skript generovat až při vlastním přihlášení.

Sdílená složka netlogon a sdílený disk pro sledování uživatelů

Sdílená složka netlogon je umístění, ze kterého si pracovní stanice se systémem Windows nahrá-vají přihlašovací skript. Tuto sdílenou složku potřebujeme pro uložení přihlašovacího skriptu, ve kterém se zajistí připojení sdíleného disku, který se využije při sledování IP adres uživatelů.

V souboru smb.conf tedy musí být něco jako:

```
logon script = netlogon.bat
```

Na tomto řádku se klientovi se systémem Windows říká, aby nahrál a provedl skript s názvem netlogon.bat. Tento skript musí být uložen ve sdílené složce netlogon. Dále potřebujeme vlast-ní skript netlogon.bat. Pro tyto účely můžete využít následující příklad, který se uloží do sdíle-né složky netlogon, v našem případě tedy do složky /home/samba/netlogon/NETLOGON.BAT.

```
REM NETLOGON.BAT net use z: \\linux\samba /yes
```

Po provedení tohoto skriptu se na pracovní stanici se systémem Windows provede připojení sdí-leného disku, takže bude možné díky výstupu z nástroje smbstatus sledovat jednotlivé uživatelea pracovní stanice.
Zatím je to jednoduché.Pro sledování uživatelů je nutný sdílený disk, který má v našem případě název Samba. Konfigu-raci tohoto sdíleného disku naleznete v souboru smb.conf:

```
[samba] comment = login tracking share path = /home/samba/samba browseable = No root preexec = /usr/local/bin/netlogon.sh %u %I root postexec = /usr/local/bin/netlogoff.sh %u
```

Jak asi odhadnete nebo víte (pokud jste četli dokumentaci Samby), parametry root preexeca root postexec způsobí, že Samba zadané skripty spustí vždy při připojení nebo odpojení sdí-leného disku. V našem případě se do skriptů jako parametr předává uživatelské jméno. Všimnětesi hodnoty %u v obou řádcích. Tyto skripty následně zajistí volání skriptu nebo programu, kterýprovede úpravy filtrovacích pravidel pro pakety procházející bránou.
Všimněte si, že skript netlogon.sh musí zkontrolovat, jestli daná pracovní stanice nemá sdílenýdisk pro sledování uživatelů již připojen.Následuje výpis skriptu netlogon.sh:

```
#!/bin/sh # # netlogon.sh # # použití: # netlogon.sh <uživatelské_jméno> # if [ -f /var/run/smbgate/$1 ] ; then
        exit 0 fi echo $2 > /var/run/smbgate/$1 IPTABLES='/usr/sbin/iptables' EXTIF='eth0' COMMAND='-A'
ADDRESS=`cat /var/run/smbgate/$1` GROUP=`groups $1 | gawk '// { print $3 }'` if [ -f /etc/smbgate/users/$1 ] ; then
                /etc/smbgate/users/$1 $COMMAND $ADDRESS $EXTIF else if
        [ -f /etc/smbgate/groups/$GROUP ] ; then /etc/smbgate/groups/
        $GROUP $COMMAND $ADDRESS $EXTIF
        else /etc/smbgate/users/default.sh $COMMAND $ADDRESS $EXTIF
        fi fi
```

Tento skript (netlogon.sh) se provede při přihlášení uživatele a vybere další skripty, které majíbýt spuštěny v závislosti na uživatelském jméně a členství uživatele ve skupinách. Pro účely sle-dování zapíše skript do souboru ve složce /var/run/smbgate IP adresu uživatele. Názvem sou-boru je uživatelské jméno a tento soubor bude použit později při odhlášení uživatele. IP adresa jepředána jako argument do skriptu s uživatelským jménem, který nakonec provede úpravu pravi-del firewallu.

Všimněte si, že skript netlogon.sh nejprve zkusí nalézt skript s uživatelským jménem, pokudnenalezne, zkusí nalézt skript s názvem skupiny, a pokud nenalezne ani ten, použije výchozískript s názvem default.sh. Tento mechanismus si můžete upravit dle vlastních potřeb, jen neza-pomeňte odpovídajícím způsobem upravit i návazné skripty.
Pokud je uživatel členem více skupin, může se stát, že při provádění skriptu dojde k chybě. Neměljsem čas vytvořit lepší kód.Následuje výpis skriptu netlogoff.sh:

```
#!/bin/sh
#
# netlogoff.sh
#
# usage:
# netlogoff.sh <username>
#
IPTABLES='/usr/sbin/iptables'
EXTIF='ppp0'
COMMAND='-D'
TRACKSHARE="samba"
ADDRESS=`cat /var/run/smbgate/$1`
GROUP=`groups $1 | gawk '// { print $3 }'`
NM=`smbstatus -u $1 | grep $TRACKSHARE | wc -l`
if [ $NM -gt 0 ]; then

        exit fi if [ -f /etc/smbgate/users/$1 ] ; then
                /etc/smbgate/users/$1 $COMMAND $ADDRESS $EXTIF else
          if [ -f /etc/smbgate/groups/$GROUP ] ; then /etc/smbgate/groups/
          $GROUP $COMMAND $ADDRESS $EXTIF
          else /etc/smbgate/users/default.sh $COMMAND $ADDRESS $EXTIF
```

```
fi fi rm -f /var/run/smbgate/$1
```

Skript netlogoff.sh je spuštěn při odhlášení uživatele a získá ze souboru uživatele ze složky /var/run/smbgate/ IP adresu. Tato adresa je následně předána jako argument uživatelského skriptu ze složky /etc/smbgate/users/, který provede aktualizaci firewallu do stavu, kdy uži-vatel není přihlášen.

Některé verze systému Windows, např. Windows 2000, provedou připojení sdíleného disku v prů-běhu přihlášení několikrát. To může následně způsobit několikanásobné spuštění skriptů netlo-gon.sh a netlogoff.sh. Proto může být vhodnější vytvořit skript pro kontrolu odhlášení a spouš-tět jej s využitím nástroje cron místo skriptu netlogoff.sh, který spouští Samba. Zde je příklad:

```
#!/bin/sh
# checklogout.sh
#
# použití:
# naplánované spuštění s využitím nástroje cron (cca každých 10 minut)


TRACKDIR=”/var/run/smbgate”
DIRLENGTH=${#TRACKDIR}
TRACKSHARE=”samba”
EXTIF=’eth0’
COMMAND=’-D’
if [ -d $TRACKDIR ]; then

    for n in $TRACKDIR/*; do
        [ -d $n ] && continue;
        if [ -f $n ] ; then

            IPADDRESS=`cat $n`
            USERNAME=${n:$DIRLENGTH+1}
            NMS=`smbstatus -u $USERNAME | grep $TRACKSHARE | grep $IPADDRESS | grep

v grep | wc -l`
            if [ $NMS == 0 ] ; then
                rm -f $n
                GROUP=`groups $USERNAME | gawk ‘// { print $3 }’`
                if [ -f /etc/smbgate/users/$USERNAME ] ; then

                  /etc/smbgate/users/$USERNAME $COMMAND
              $IPADDRESS $EXTIF else if [ -f /etc/smbgate/groups/
              $GROUP ] ; then /etc/smbgate/groups/$GROUP
              $COMMAND $IPADDRESS $EXTIF
              else /etc/smbgate/users/default.sh $COMMAND
              $IPADDRESS $EXTIF fi fi else exit 0 fi done fi
```

Při použití tohoto skriptu je nutné z konfigurace sdíleného disku pro sledování uživatelů v sou-boru smb.conf odstranit parametr root postexec:

```
root postexec = /usr/local/bin/netlogoff.sh %u
```

Následuje standardní skript pro jednotlivého uživatele ze složky /etc/smbgate/users. Tento skript provede úpravu pravidel firewallu: #!/bin/sh # COMMAND=$1 ADDRESS=$2 EXTIF=$3 IPTABLES=’/usr/sbin/iptables’ $IPTABLES $COMMAND POSTROUTING -t nat -s $ADDRESS -o $EXTIF -j MASQUERADE

Ve složce /etc/smbgate/users/ je nutné mít i skript default.sh, který zajistí výchozí chování brány:

#!/bin/sh # # default.sh COMMAND=$1 ADDRESS=$2 EXTIF=$3 IPTABLES=’/usr/sbin/iptables’ #$IPTABLES $COMMAND POSTROUTING -t nat -s $ADDRESS -o $EXTIF -j MASQUERADE exit 0

Alternativní řešení

Celý princip řešení, založený na připojování sdíleného disku pro sledování uživatelů, spouštění skriptů pro aktualizaci pravidel firewallu a čekání na jejich další spuštění při odpojení sdíleného disku pro sledování uživatelů a reset pravidel firewallu se může někomu zdát překombinovaný a chabý. Celý princip se může také změnit v případě, kdy vývojáři Samby přidají nové vlastnosti.

Poslední verze Samby již umožňuje vytvořit výpis přihlášených uživatelů. Sám jsem tuto vlastnost využil ve skriptu, který provádí sledování uživatelů a aktualizaci pravidel firewallu při přihlášení a odhlášení uživatelů. Tento skript nevyžaduje tolik

práce jako řešení v tomto návodu a je velmi jednoduchý.

Veškerý kód si můžete stáhnout na adrese: *http://sourceforge.net/projects/smbgate/.*

Nastavení SSH

Někdy můžete chtít nechat spuštěný primární řadič domény na jednom počítači a jako bránu pou-žívat jiný počítač. Pak je nutné nastavit bránu tak, aby přijímala přihlášení autentizovaná s pomo-cí rsa bez hesel z primárního řadiče. Více informací o nastavení serveru a klienta ssh naleznete zde: *www.openssh.org.*

> Důležité
>
> Před nastavením rsa nebo jakéhokoliv jiného druhu kryptografické autentizace byste si měli přečíst příslušnou dokumentaci. Pokud zabezpečení není problémem, použijte můj příklad a pokračujte.

Generování klíčů

Pro vygenerování klíčů proveďte na počítači, který má být primárním řadičem domény, následující příkazy:
pdc:~# ssh-keygen -t rsa

Odpovězte na všechny otázky a výsledný veřejný klíč zkopírujte na počítač, který slouží jako brána. Veřejný klíč bývá obvykle uložen v souboru "~/.ssh/id_rsa.pub".

pdc:~# cd .ssh pdc:~# scp id_rsa.pub root@gateway:/root/.ssh/authorized_keys2

Přihlašovací skript s podporou SSH

Následuje výpis uživatelského skriptu ze složky /etc/smbgate/users/ s podporou zabezpečené autentizace ssh:

#!/bin/sh # COMMAND=$1 ADDRESS=$2 EXTIF=$3 IPTABLES='/sbin/iptables' ssh root@gateway "$IPTABLES $COMMAND POSTROUTING -t nat -s $ADDRESS -o $EXTIF j MASQUERADE"

Všimněte si, že spustitelný soubor iptables se volá s využitím ssh na počítači s názvem „gateway".

Nastavení pracovní stanice se systémem Windows

Úvod

V této části naleznete informace o nastavení sítě, správy uživatelů a zásad zabezpečení na pra-covní stanici se systémem Windows. Naopak zde nenaleznete úplný popis včetně přesných názvů všech dialogových oken, protože předpokládám, že pokud čtete a rozumíte tomuto návodu, tak zvládnete i toto nastavení.

Síové protokoly

Nejprve, pokud je doopravdy nepotřebujete, odstraňte všechny síťové protokoly kromě TCP/IP. Dokonce i bez jejich vlastního protokolu mají stanice se systémem Windows v oblibě vysílat do sítě hodně paketů se všeobecnými adresami – a to nikoho netěší. A kdo potřebuje jiný další protokol kromě TCP/IP?

Nastavení DHCP

Při konfiguraci serveru DHCP na linuxovém počítači je nutné nakonfigurovat server DHCP tak, aby přiděloval klientským počítačům se systémem Windows kromě vlastní adresy IP i adresy jmen-ných serverů a adresu výchozí brány. Díky tomu nebude nutné nastavovat tyto údaje na každé pracovní stanici zvlášť.

Připojení do domény realizované linuxovým serverem

Klientskou stanici připojte do domény a jako název domény zadejte název serveru se systémem Linux. To je pro správné fungování brány zásadní. Při připojení některých verzí systému Windows k řadiči domény, realizované Sambou na linuxo-vém počítači, je nutné na řadiči domény vytvořit účty počítačů. Podrobnější informace o tom, jak nastavit Sambu jako primární řadič domény pro konkrétní verze systému Windows, který použí-váte, naleznete v dokumentaci k Sambě.

Windows for Workgroups

Tato verze nemá na konfiguraci připojení do domény realizované linuxovým serverem žádnézvláštní požadavky.Přihlašovací skript by měl mít název WfWg.bat tak, aby byl spuštěn po vyhodnocení parametru %a.Příklad:

REM WFWG.BAT net use z: \\linux\samba /yes

Windows 95/98/ME

Tyto verze pro připojení do domény realizované linuxovým serverem také nevyžadují žádnou spe-ciální konfiguraci.Přihlašovací skript by měl mít název W95.bat tak, aby byl spuštěn po vyhodnocení parametru %a.Příklad:

```
REM W95.BAT net use z: \\linux\samba /yes
```

Windows NT

Tato verze vyžaduje vytvoření účtu počítače na linuxovém serveru. Podrobnosti naleznete v doku-mentaci k Sambě. Přihlašovací skript by měl mít název WinNT.bat tak, aby byl spuštěn po vyhod-nocení parametru %a.

Příklad:

```
REM WINNT.BAT net use z: \\linux\samba /yes /persistent:no
```

Windows 2000

Tato verze vyžaduje vytvoření účtu počítače na linuxovém serveru. Podrobnosti naleznete v doku-mentaci k Sambě. Přihlašovací skript by měl mít název Win2K.bat tak, aby byl spuštěn po vyhod-nocení parametru %a.

Příklad:

```
REM WIN2K.BAT net use z: \\linux\samba /yes /persistent:no
```

Windows XP

Tato verze vyžaduje vytvoření účtu počítače na linuxovém serveru a úpravu registru. Úprava registru spočívá v nastavení hodnoty 0 pro klíč „HKEY_LOCAL_MACHINE\SYSTEM\CurrentContro-lSet\Services\Netlogon\Parameters\RequireSignOrSeal“. Výchozí hodnota tohoto klíče je
1. Po nastavení na hodnotu 0 nebudou s připojením do domény žádné problémy. Pokud je nutné nastavení provést u více pracovních stanic, je možné vytvořit soubor s libovolným názvem a příponou .reg, který má následující obsah:

```
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Netlogon\Parameters] “requiresignorseal”=dword:00000000
```

Tato verze systému Windows také vyžaduje úpravu přihlašovacího skriptu, protože někdy vytvá-ří připojení jako trvalé. Přihlašovací skript by měl mít název WinXP.bat tak, aby byl spuštěn po vyhodnocení parametru %a.

Příklad:

```
REM WIN2K.BAT net use z: \\linux\samba /yes /persistent:no
```

Editor skupinové politiky

Součástí instalačního disku systému Windows je nástroj na úpravu uživatelské politiky – *Policy edi-tor*. Název spustitelného souboru je poledit.exe. Tento nástroj, jak jeho název napovídá, je možné použít pro vytvoření uživatelské i systémové politiky. Nástroj však bohužel negeneruje žádný výstup ve formátu prostého textu, takže zde není uveden žádný příklad.

Policy editor použijte pro vytvoření zásad pro pracovní stanice a uživatele. V nastavení by mělo být potlačeno místní ukládání hesel a ukládání doménových hesel pro zvýšení úrovně zabezpe-čení. Soubor uložte pod názvem config.pol a umístěte jej do sdílené složky netlogon na linu-xovém serveru. Klientské stanice si tento soubor nahrají a automaticky aplikují v něm uvedené nastavení.

Pokud soubor config.pol nepoužijete, pracovní stanice se systémem Windows budou obtěžovat se žádostmi o zadání hesla do Windows a synchronizace a správa hesel do systému Windows i doménových hesel bude obtížná.

Správa uživatelů a skupin

Přidání uživatelů

Přidání uživatele systému Linux obvykle znamená nastavení hesla pro Sambu s využitím nástroje smbpasswd. Pokud máte pochybnosti, podívejte se do dokumentace k Sambě. To není velký pro-blém.

Správa hesel

Toto téma jsem do návodu vložil proto, že neznám způsob, jak spravovat uživatele a jejich hesla na pracovní stanici se systémem Windows bez použití webového rozhraní. Nemůžu najít žádné nástroje, které by pomohly tento problém řešit. Proto pro tyto účely používám CGI skript.

Vyzkoušejte balík, který naleznete na adrese *http://changepassword.sourceforge.net*, vypadá jako dobrá volba.

Přidělení nebo odepření přístupu uživatelům

Jak bylo vidět v předchozí části tohoto návodu, démon Samby spustí skript netlogon.sh při kaž-dém připojení sdíleného disku pro sledování uživatelů. Tento skript volá další skript, jehož název obsahuje uživatelské jméno, kterému jako parametr předá IP adresu pracovní stanice, ze které se uživatel přihlašuje. Tento uživatelský skript následně aplikuje všechna potřebná pravidla firewallu.

Pokud chcete přidělit uživateli úplný přístup k Internetu, může skript vypadat například takto:

```
#!/bin/sh # COMMAND=$1 ADDRESS=
$2 EXTIF=$3
IPTABLES='/usr/sbin/iptables'
$IPTABLES $COMMAND POSTROUTING -t nat -s $ADDRESS -o $EXTIF -j MASQUERADE
```

Pokud pro daného uživatele nechcete provést žádnou změnu pravidel, vytvořte pro něj prázdný skript:

```
#/bin/sh # exit 0
```

Pro uživatele s nižší úrovní oprávnění nemusíte vytvářet žádný skript, pak bude zajištěno spuště-ní skriptu default.sh, který může být buď prázdný nebo může uživateli povolit pouze omezený přístup takto:

```
#!/bin/sh # COMMAND=$1 ADDRESS=$2 EXTIF=$3 EXTIFADDRESS=$4 IPTABLES='/usr/sbin/iptables' $IPTABLES $COMMAND
POSTROUTING -t nat -s $ADDRESS -o $EXTIF —dport 25 -j SNAT — to-source $EXTIFADDRESS $IPTABLES $COMMAND
POSTROUTING -t nat -s $ADDRESS -o $EXTIF —dport 110 -j SNAT — to-source $EXTIFADDRESS
```

Pokud chcete do tohoto skriptu přidat další parametr, je nutné upravit všechny předchozí skripty. Celý návod vám však nebude k ničemu užitečný, pokud nerozumíte syntaxi nástroje iptables.

Vytváření skupin

Zde stačí příslušné skupiny vytvořit přímo na primárním řadiči domény na linuxovém systému

a přidat do skupin příslušné uživatele. To je vše.Nezapomeňte, že skripty z uvedeného příkladu se neprovedou správně v případě, kdy uživatelpatří do několika skupin. Pokud takové uživatele máte, je nutné skripty upravit.

Zásady skupin

V tomto případě je nutné definovat skripty pro skupiny a umístit je do složky „/etc/smbgate/groups/“. Nezapomeňte, že skript musí mít stejný název jako skupina. Logika ve zdrojovém kódu se nejprve snaží nalézt skript s názvem uživatele, pak skript s názvem skupiny a nakonec (pokud ani uživatelský, ani skupinový skript neexistuje) výchozí skript. Pokud budete chtít upravit toto chování, upravte příslušným způsobem skripty netlogon.sh, netlogoff.sh nebo checklogout.sh.

Bezpečnost

Úvod

Tento dokument hovoří o hlavních bodech týkajících se bezpečnosti linuxových systémů. Popi
suje obecné postupy a odkazuje se na další informační prameny.S problematikou bezpečnosti souvisí i řada dalších praktických návodů, na které se v případěpotřeby odkazujeme.

Tento dokument nemá za cíl představovat aktuální rizika. Nová rizika se objevují velmi často. Zde se dočtete, kde takovéto aktuální informace získat, a dále se dozvíte o obecných postupech, jak řadu rizik eliminovat.

Veškeré komentáře, upozornění na chyby doplňující informace a kritiky všeho druhu zasílejte na adresy autorů návodu: kevin-securityhowto@tummy.com a dave@linuxsecurity.com.

Poznámka

Prosíme, abyste posílali připomínky *oběma* autorům a jako předmět zprávy uváděli

„Linux“, „security“ nebo „HOWTO“, aby nebyly tyto zprávy odfiltrovány jako spamy.

Řada příkladů a popisů se vztahuje k systému a k balíkům firmy RedHat(tm). V jiných systémech se mohou projevovat jinak.

Přehled

Proč potřebujeme bezpečnost?

Ve stále se měnícím světě globálních datových komunikací, levných internetových linek a rychlé-ho vývoje programů se bezpečnost stává stále větším a větším problémem. Bezpečnost předsta-vuje základní požadavek, protože globální komunikace ze své podstaty bezpečná není. Při pře-nosu dat mezi místy A a B na Internetu mohou data procházet celou řadou uzlů, kde mají jiní uži-vatelé možnost data vidět, případně i modifikovat. I ostatní uživatelé vašeho systému mohou záměrně modifikovat vaše data způsobem, který si nepřejete. Neautorizovaný přístup k vašemu systému může získat útočník (označovaný také jako „cracker“), který se pak může za vás vydávat, odcizit vaše data, nebo vám dokonce zabránit v přístupu k vlastnímu systému. Pokud vás zajímá, jaké jsou rozdíly mezi „hackerem“ a „crackerem“, doporučujeme vám dokument Erika Raymonda „How to Become a Hacker“, který je dostupný na adrese *http://www.tuxedo.org/~esr/faqs/ hacker-howto.html.*

Jak bezpečné je bezpečné?

Je nutné mít na paměti, že žádný počítačový systém nemůže být úplně bezpečný. Jediné, čeho můžete dosáhnout, je to, aby bylo narušení systému různě obtížné. U běžného počítačového uži-vatele nejsou zapotřebí žádná speciální opatření k zabezpečení systému. U náročných uživatelů (banky, telekomunikační firmy a podobně) je zapotřebí vyšší míra zabezpečení.

Dále je třeba vzít v úvahu, že čím je systém bezpečnější, tím nepříjemnější jsou příslušná bezpeč-nostní opatření. Je nutné se rozhodnout, kde je ona hranice, při níž je systém dostatečně bezpeč-ný a zároveň stále rozumně použitelný. Můžete například požadovat, aby u všech uživatelů, kteří se k vašemu systému připojují telefonicky, bylo provedeno zpětné volání na jejich telefonní číslo. Je to bezpečnější než přímé přihlašování z jakéhokoliv čísla, ale komplikuje to život uživatelům, kteří se zrovna nepřipojují z domova. Můžete počítač nainstalovat bez síťového rozhraní a bez při-pojení k Internetu, tím se ale limituje jeho použitelnost.

U středních a větších sítí je rozumné vytvořit bezpečnostní politiku, která definuje, jaká míra bez-pečnosti je požadována a jakým způsobem se kontroluje. Známý příklad bezpečnostní politiky můžete najít v RFC 2196. Tento dokument byl nedávno aktualizován a představuje vynikající základ pro vytvoření vaší vlastní bezpečnostní politiky.

Co se snažíte chránit?

Než začnete svůj systém zabezpečovat, měli byste stanovit, proti jakému ohrožení jej chcete chránit, která rizika jste a nejste ochotni nést a jaká bude výsledná zranitelnost systému. Měli byste provést analýzu systému, abyste věděli, co chráníte, proč to chráníte, jakou to má hodnotu a kdo je zodpovědný za data a další hodnoty.

- *Riziko* představuje možnost, že by útočník mohl úspěšně získat přístup k vašemu systému. Může pak takový útočník číst a zapisovat soubory? Může spouštět programy, které by mohly vyvolat nějakou škodu? Může smazat kritická data? Může vám nebo vaší firmě zabrá-nit v plnění důležitých úkolů? Nezapomínejte také, že pokud někdo získá přístup k vaše-mu účtu a k vašemu systému, může se také vydávat za vás.
  Navíc, jakmile dojde k napadení jednoho účtu na jednom počítači, může následně dojít k napadení celé sítě. Pokud například povolujete uživatelům přihlašovat se pomocí sou-borů .rhosts nebo používáte nezabezpečené služby jako tftp, pootevíráte tím útočníkovi dveře. Jakmile se mu podaří získat přístup k jednomu účtu na některém systému, může jej následně použít k získání přístupu k jiným účtům a k jiným systémům.
- *Ohrožení* zpravidla pochází od někoho, kdo má nějakou motivaci získat neautorizovaný přístup k vaší síti nebo počítači. Musíte uvážit, komu chcete přístup umožnit a jaké ohro-žení to může představovat.

  Existuje několik typů útočníků a při zabezpečování systému je rozumné mít na paměti jejich charakteristiky:

Zvědavec – Tento typ útočníka se typicky zajímá, co je váš systém zač a jaká data obsahuje.
Záškodník – Tento typ útočníka se snaží buď váš systém vyřadit z činnosti nebo poško-dit vaše webové stránky nebo vás prostě nějakým jiným způsobem donutit vynaložit čas a prostředky na nápravu jím způsobené škody.
Ambiciózní útočník – Tento typ se snaží napadením vašeho systému získat popularitu. Napadá exponované systémy, aby dokázal své schopnosti.
Konkurent – Tento typ útočníka se zajímá o data, která v systému máte. Může se domnívat, že máte něco, co mu může přinést prospěch, ať finanční nebo jiný.
Vypůjčovatel – Tento typ útočníka má zájem využít váš systém a jeho prostředky ke svým účelům. Typicky chce provozovat chatové servery nebo servery irc, archivy pornostránek, nebo dokonce servery DNS.
Skokan – Tento útočník se váš systém snaží použít pouze jako prostředek k napadení dalších systémů. Pokud má váš systém dobré připojení, nebo slouží jako brána k dalším systémům, můžete očekávat tento typ útoků.

- *Zranitelnost* udává, jak dobře je váš počítač chráněn před jinými systémy a jaká je možnost získat k němu neautorizovaný přístup. Co je v sázce, jestliže se někdo do systému nabou-rá? Samozřejmě se to liší u domácího uživatele připojeného modemem a u společnosti s počítači připojenými k Internetu a k jiným sítím.

Kolik času vám zabere obnovení ztracených dat? Počáteční časová investice na zabezpečení vám může ušetřit deseti i vícenásobek času později při obnovování dat. Pořizujete pravidelně zálohy a kontrolujete jejich funkčnost?

Vytvoření bezpečnostní politiky

Vytvořte jednoduchou obecnou bezpečnostní politiku, kterou budou vaši uživatelé znát a budou se jí řídit. Měla by chránit jak data, tak i soukromí uživatelů. Další možnosti, které stojí za zváže-ní, jsou: Kdo má mít přístup k systému (může můj účet používat můj kamarád?), kdo je oprávněn instalovat programy, kdo vlastní která data, jak provádět obnovu v případě poškození a jaké je akceptovatelné užití systému.

Obecně přijímané bezpečnostní politiky začínají větou:

„Co není povoleno, je zakázáno.“

Znamená to, že pokud uživateli nepovolíte přístup ke službě, uživatel ji nesmí použít. Zajistěte,

aby politika fungovala i pro běžné uživatelské účty. Přístup „toto nejsem schopen jednoduše udě-lat, udělám to jako *root*“ může vést ke vzniku zjevných bezpečnostních děr, které ani nemusí být v daném okamžiku využitelné.
Dokument RFC1244 popisuje, jak vytvořit vlastní síťovou bezpečnostní politiku.*Dokument RFC1281* představuje příklad bezpečnostní politiky s podrobným vysvětlením. Koneč-ně se můžete podívat do archivu politik na adrese ftp://coast.cs.purdue.edu/pub/doc/policy, kdenajdete příklady skutečných bezpečnostních politik.

Opatření k zabezpečení systému

Tento dokument popisuje různá opatření k zabezpečení hodnot, na nichž jste tvrdě pracovali: zabezpečení počítače, dat, uživatelů, sítě, dokonce i reputace. Jak dopadne vaše pověst, pokud útočník smaže data nějakého uživatele? Nebo pokud změní vaše webové stránky? Nebo zveřejní plány vaší společnosti na příští čtvrtletí? Pokud plánujete instalaci sítě, musíte vzít v potaz celou řadu faktorů ještě předtím, než k síti připojíte jediný počítač.

I pokud používáte pouze modemové připojení nebo provozujete malou síť, neznamená to, že se o vás nějaký útočník nebude zajímat. Cílem útoků nejsou pouze populární a velké systémy – řada útoční-ků se prostě snaží napadnout co největší počet systémů bez ohledu na jejich velikost. Navíc mohou bezpečnostní díru ve vašem systému použít pro přístup k dalším systémům. Útočník má dostatek času a může si dovolit hádat, jak jste systém zabezpečili, prostým zkoušením všech možností. Existuje i řada jiných důvodů, proč může být váš systém cílem útoku, budeme o nich hovořit později.

Zabezpečení počítače

Jde o oblast zabezpečení, na kterou se správce většinou soustřeďuje. Typicky to obnáší zajistit bez-pečnost svého systému a předpokládat, že ostatní dělají totéž. Mezi činnosti, za které je správce systému zodpovědný, patří volba dobrých hesel, zabezpečení síťových služeb, údržba záznamů
o činnosti počítače a aktualizace programů se známými bezpečnostními chybami. I když jde
o naprosto nezbytné činnosti, mohou být velmi únavné, jakmile se síť rozroste na více než jen několik počítačů.

Zabezpečení sítě

Bezpečnost sítě je stejně nutná jako bezpečnosti, počítače. Pokud máte v síti stovky nebo tisíce počítačů, těžko můžete očekávat, že každý jeden bude dostatečně zabezpečen. Zajistíte-li přístup pouze autorizovaným uživatelům, nainstalujete firewally, použijete silné šifrování a zajistíte, že v síti nejsou žádné nezabezpečené počítače, zvyšujete bezpečnost celé sítě. V tomto dokumentu popisujeme některé techniky k zabezpečení sítě a snad vám ukážeme některé způsoby, jak útoč-níkům zabránit v

přístupu k tomu, co se snažíte chránit.

Zabezpečení utajením

Jedním ze způsobů zabezpečení, o kterém se musíme zmínit, je „zabezpečení utajením“. Zname-ná to například, že službu se známou bezpečnostní chybou spustíme na nějakém nestandardním portu a budeme doufat, že útočník si její existence nevšimne a nevyužije ji. Buďte si jisti, že útoč-níci budou schopni službu najít a využít. Zabezpečení utajením nepředstavuje žádné zabezpeče-ní. Jen proto, že se staráte o malou nebo neznámou síť, neznamená to, že se vám útoky vyhnou.

V dalších kapitolách si popíšeme, co se snažíme chránit.

Organizace tohoto návodu

V tomto návodu se budeme zabývat některými běžnými problémy z oblasti zabezpečení. Dozvíte se:

jak počítač bránit před fyzickým nebezpečím;
jak zabezpečit systém před autorizovanými uživateli;
jak nastavit souborové systémy a přístupová práva k souborům;
jak pomocí šifrování zlepšit bezpečnost počítače a sítě;
které parametry jádra mohou ovlivnit bezpečnost;
jak zabezpečit systém před útoky po síti;
jak počítač připravit před připojením k síti;
co dělat, jakmile detekujeme probíhající nebo provedený útok.

Samozřejmě nebude chybět přehled důležitých zdrojů, souhrn obvyklých otázek a odpovědía závěrečné shrnutí.Dvě hlavní zásady, které byste si měli z tohoto dokumentu odnést, jsou:

Hlídejte svůj systém. Sledujte logy, jako například /var/log/messages.
Udržujte systém aktuální, instalujte nové verze programů a sledujte bezpečnostní upozornění. Už jenom tímto výrazně zvýšíte bezpečnost systému.

Fyzická bezpečnost

První úroveň zabezpečení systému je jeho fyzická bezpečnost. Kdo má k počítači přímý fyzický

přístup? A má jej mít? Jste schopni počítač ochránit před jejich zásahy? Míra fyzického zabezpečení je silně závislá na typu systému a/nebo na prostředcích, které jsouk dispozici.

Pokud jde o domácí počítač, pravděpodobně jej nepotřebujete příliš chránit (i když by měl býtchráněn před zásahy dětí a otravných příbuzných). Pokud jde o počítač například v učebně, musí-te jej zabezpečit lépe, nicméně stále musíte nechat uživatelům možnost jej použít. U počítačev kanceláři můžete uvažovat o jeho zabezpečení v mimopracovní době nebo jste-li pryč. U někte-rých společností může být opuštění a nezabezpečení terminálu důvodem k výpovědi. Zjevná opat-ření pro fyzické zabezpečení jsou zámky na dveřích a kabelech, zamykatelné skříně, instalacekamerového systému a podobně. To už je ale mimo záběr tohoto dokumentu.

Zámky počítačů

Většina moderních počítačů umožňuje nějakou míru uzamčení. Obvykle je na čelním panelu zámek, kterým můžete zamknout skříň počítače, a nikdo tak nebude moci přímo manipulovat s hardwarem počítače. V některých případech se tak dá zabránit nabootování počítače z vlastní diskety nebo disku.

Zámky na skříních podle své konstrukce mohou plnit různé funkce. V některých případech pouze neumožňují otevřít skříň počítače bez násilí. V jiných případech mohou zabránit dokonce i v při-pojení vlastní klávesnice nebo myši. Podrobnosti naleznete v manuálu k počítači. V některých pří-padech mohou být tyto zámky velmi užitečné, i když jejich kvalita je mnohdy mizerná a k jejich „odemčení“ stačí šroubovák.

Některé počítače (hlavně SPARC a MAC) mají zezadu skříně očko, které je možné použít k zaplombování nebo zamčení skříně.

Bezpečnost BIOSu

BIOS je nízkoúrovňový software, který slouží ke konfiguraci hardwarových prvků počítačů na platformě x86. Služby BIOSu využívá LILO i jiné zavaděče Linuxu ke zjištění, jak systém zavést. I na jiných platformách, na nichž Linux běží, existuje podobná vrstva (Open Firmware na Macu a nových Sunech, bootovací PROM na Sunech a podobně). Pomocí služeb BIOSu můžete útoční
kovi zamezit v restartu počítače a manipulaci se systémem. Řada BIOSů umožňuje nastavit heslo pro spuštění počítače. To sice

moc neřeší (BIOS je možné vymazat nebo změnit, pokud má útočník přístup do skříně počítače), může však fungovat jako dobré odstrašující opatření (zpomaluje to a zanechává stopy po napadení). Podobně i na počíta-čích platformy Sparc je možné nastavit EEPROM tak, aby při spuštění systému bylo vyžadováno heslo. To může útočníka zpomalit.

Dalším rizikem při použití hesla do BIOSu jsou implicitní hesla. Většina výrobců BIOSů nepředpokládá, že uživatel v případě zapomenutí hesla bude otevírat počítač a odpojovat baterii, a proto jejich BIOSy obsahují implicitní hesla, která fungují vždy. Mezi známá implicitní hesla patří:

j262, AWARD_SW, AWARD_PW, lkwpeter, Biostar, AMI, Award, bios, BIOS, setup, cmos, AMI!SW1, AMI?SW1, password, hewittrand, shift + s y x z Zkoušel jsem Award BIOS a fungovalo heslo AWARD_PW. Tato hesla je možné snadno zjistit na stránkách výrobců BIOSu nebo na adrese *http://astalavista.box.sk.* Proto je nelze považovat za ochranu před znalým útočníkem.

Řada BIOSů umožňuje nastavit i další rozumná bezpečnostní opatření. Podívejte se do manuálu k počítači nebo přímo do BIOSu při startu počítače. Existují například možnosti zakázat bootová-ní počítače z diskety nebo ochrana některých funkcí heslem.

Poznámka

Pokud nastavíte heslo do BIOSu na serveru, nebudete jej schopni spustit bez fyzického zásahu. Budete k němu muset zajet a zadat heslo například i po výpadku proudu.

Zabezpečení zavaděče systému

Většina zavaděčů Linuxu umožňuje také zadat heslo. LILO například obsahuje volby password a restricted. Volba password vyžaduje zadání hesla při každém spuštění systému, volba restricted *pouze v případě spouštění s parametry (například single)*.

Uvádíme výtah z manuálové stránky lilo.conf:

Globální parametry password = heslo

Stejně jako volba password jednoho obrazu (viz níže), platí pro všechny spouštěné obrazy. restric-ted Stejně jako volba restricted jednoho obrazu (viz níže), platí pro všechny spouštěné obrazy.

Parametry obrazů password = heslo

Spuštění daného obrazu je chráněno heslem. restricted

Heslo se vyžaduje pouze při pokusu spustit obraz s explicitně zadanými parametry (např. single).Při nastavování všech těchto hesel nezapomínejte, že si je budete muset pamatovat. Rovněž neza-pomínejte, že tato hesla mohou zkušeného útočníka pouze zpomalit. Nezabrání mu totiž v nabo-otování systému z diskety a následném připojení disku. Pokud chcete zavaděč chránit heslem,měli byste zakázat bootování z diskety a chránit heslem i BIOS.Nezapomeňte, že soubor /etc/lilo.confmusí mít nastavena práva „600“ (tedy čitelný pouze pro superuživatele), jinak z něj kdokoliv bude moci hesla přečíst.

Z informační stránky GRUB:

GRUB vyžaduje heslo, takže interaktivní operace může spustit pouze správce (tj.editaci položek v nabídce a zadávání rozhraní příkazových řádků). Abyste mohlituto vlastnost využít, musíte použít příkaz password v konfiguračním souboru,například:

password --md5 PASSWORD

Je-li zadán, GRUB zablokuje veškeré interaktivní řízení, dokud nestisknete klá-vesu p a nezadáte správné heslo. Volba --md5 znamená, že heslo je ve tvaru MD5.Není-li tato volba uvedena, GRUB očekává heslo ve tvaru prostého textu.Heslo můžete zašifrovat pomocí příkazu md5crypt, a to například tak, že spustí-te grub shell a zadáte heslo:

grub> md5crypt Password: ********** Encrypted: 1U$JK7xFegdxWH6VuppCUSIb.

a pak jej přenesete do konfiguračního souboru.

Grub má taktéž „uzamykací“ příkaz, kterým můžeme zamknout oblast, když není zadáno správné heslo. Přidejte pouze „lock“ a oblast se znepřístupní do té doby, než uživatel zadá správné heslo. Uvítáme informace o zabezpečení jiných zavaděčů (grub, silo, milo, linload a podobně).

Poznámka

Pokud nastavíte heslo zavaděče na serveru, nebudete jej schopni spustit bez fyzického zásahu. Budete k němu muset zajet a zadat heslo například i po výpadku proudu.

xlock a vlock

Pokud se občas od počítače na chvíli vzdálíte, budete možná chtít „zamknout“ konzolu, aby nikdo nemohl zasahovat do vaší práce nebo si ji prohlížet. K tomuto účelu můžete použít dva progra-my: xlock a vlock.

Program xlock zamyká grafickou konzolu a měl by být součástí každé distribuce s podporou systému X Window. Podrobnosti o tomto programu zjistíte na jeho manuálové stránce, obecně jej však můžete spustit z kteréhokoliv grafického terminálu. Program zamkne displej a k jeho ode-mknutí je nutné zadat heslo.

Program vlock je jednoduchý malý program, který umožňuje zamknout některé nebo všechny vir-tuální konzoly. Můžete zamknout buď tu, na níž právě pracujete, nebo všechny. Pokud zamkne-te jen jednu konzolu, kdokoliv bude moci k počítači přijít a použít jej, nebude však mít přístup k vaší virtuální konzole. Je součástí ditribuce RedHat, názory na jeho užitečnost se mohou různit.

Zamknutím konzoly sice zabráníte tomu, aby někdo manipuloval s vaší prací, nezabráníte mu však počítač restartovat. Nezabráníte tím také v přístupu k počítači po síti. Důležité také je, že tím nezabráníte nikomu v přepnutí se z grafického prostředí a v práci na vir-tuální textové konzole nebo na konzole, z níž byl systém X Window spuštěn – pak jej může zasta-vit a získat vaše práva. Z toho důvodu byste měli zamykání grafické konzoly používat pouze v pří-padě, že je pod správou xdm.

Bezpečnost lokálních zařízení

Pokud máte k počítači připojenu webkameru nebo mikrofon, měli byste zvážit, zda neexistuje nebezpečí, že útočník k těmto zařízením získá přístup. Pokud je zrovna nepoužíváte, je nejrozumnější je odpojit. V opačném případě byste měli pozorně prověřit programy, které umožňují k těmto zařízením přístup.

Detekce fyzického narušení počítače

První věc, které byste si měli vždy všimnout, je reboot počítače. Protože Linux je robustní a sta-bilní operační systém, mělo by k jeho restartu docházet pouze v případě, že jste prováděli aktua-lizaci systému, hardwaru a podobně. Pokud došlo k restartu počítače bez vašeho zásahu, může tosignalizovat pokus o jeho napadení. Řada metod napadení počítače vyžaduje jeho restart nebovypnutí.

Hledejte známky zásahu do skříně počítače. I když řada útočníků maže záznamy o své činnosti

ze systémových logů, je dobré je zkontrolovat a hledat něco podezřelého. Rozumné je rovněž ukládat logovací soubory na bezpečném místě, například na vyhrazeném logo-vacím serveru na chráněné síti. Jakmile dojde k narušení počítače, jsou mnohdy k ničemu i logo-vací záznamy, protože byly s velkou pravděpodobností také změněny.

Démon *syslog* je možné nakonfigurovat tak, aby automaticky odesílal data na centrální logovacíserver. Data však při přenosu nejsou šifrována a případný útočník by je mohl vidět a dozvědět setak citlivá data. Existují verze démona, které umožňují data při přenosu šifrovat.
Nezapomínejte také, že záznamy v logu je možné snadno zfalšovat. Syslog dokonce přijme po sítidata tvářící se jako z lokálního systému, aniž by to dal najevo.Některé věci, na něž byste si měli dávat pozor:

Krátké a neúplné záznamy.
Záznamy s divným časem.
Záznamy s nesprávnými právy nebo vlastnictvím.
Záznamy o restartech služeb.
Chybějící záznamy.
Přihlášení a su z neobvyklých míst.

O datech v logovacích souborech budeme mluvit později v části nazvané „Sledujte logy systému“.

Lokální bezpečnost

Další věcí, kterou je nutné k zabezpečení systému sledovat, je zabezpečení před útoky lokálních uživatelů. Že jsme řekli *lokální* uživatelé? Ano! Získání přístupu k lokálnímu uživatelskému účtu je první věc, kterou obvykle útočník dělá, aby získal práva superuživatele. Při nedostatečném lokálním zabezpečení je možné normální uživatel-ský účet „povýšit“ na superuživatele s využitím různých chyb a nevhodně nastavených služeb. Pokud bude systém dobře lokálně zabezpečen, bude mít útočník obtížnější pozici. Dokonce i lokální uživatel může v systému napáchat dost škody, i když je opravdu tím, za koho se vydá-vá. Vytvářet účty lidem, které neznáte nebo u nichž nemáte kontaktní informace, není příliš

rozumné.

Vytváření nových účtů

Měli byste zajistit, že uživatelské účty poskytují pouze ta minimální oprávnění, která uživatelé ke své práci potřebují. Pokud pro svého desetiletého syna vytvoříte účet, budete chtít, aby mohl pouze psát a malovat, ale nemohl mazat cizí data.

Několik základních pravidel, pokud umožňujete přístup ke svému systému i dalším uživatelům:
Poskytněte jim minimální potřebná oprávnění.
Sledujte, kdy a odkud se přihlašují nebo by se měli přihlašovat.
Nezapomínejte mazat nepoužívané uživatelské účty, což můžete zjistit příkazem last nebo ze systémových logů.
Doporučuje se na všech počítačích v síti používat stejná uživatelská jména, protože se tím zjednodušuje správa účtů a analýza logů.
Absolutně se nedoporučuje vytváření skupinových uživatelských účtů. Individuální účty je možné snáze monitorovat, což u skupinových nelze.

Účty používané k útokům jsou často dlouhodobě nepoužívané. Protože je nikdo nepoužívá, jsou optimální jako nástroje k útoku.

Bezpečnost superuživatele

Nejdůležitějším uživatelským účtem je superuživatelský účet. Tento uživatel má plnou vládu nad počítačem, případně dalšími počítači v síti. Superuživatelský účet byste měli vždy používat jen krátce, pro konkrétní úkony, a jinak pracovat jako běžný uživatel. Dokonce i drobná chyba pro-vedená superuživatelem může mít vážné důsledky. Čím méně superuživatelský účet používáte, tím jste bezpečnější.

Uveďme si několik triků, jak coby superuživatel nezlikvidovat vlastní počítač:

Při provádění složitějších operací je zkuste nejprve provést nedestruktivním způsobem. Například pokud chcete provést rm foo*.bak, zkuste nejprve ls foo*.bak, abyste viděli, zda mažete skutečně ty soubory, které chcete. Vhodné je také namísto destruktivních pří-kazů použít příkaz echo.
Implicitně používejte alias k příkazu rm, který bude žádat potvrzení při mazání souborů.
Jako superuživatel provádějte pouze jednotlivé konkrétní úkony. Pokud se přistihnete, že něco zkoumáte a zkoušíte, vraťte se do normálního režimu do doby, než budete přesně vědět, co jako superuživatel udělat.
Nesmírně důležitá je příkazová cesta superuživatele. Příkazová cesta (tedy hodnota vnější proměnné PATH) udává, ve kterých adresářích má příkazový interpret hledat příkazy. Pro superuživatele by měla být cesta co nejkratší a nikdy by neměla obsahovat . (tedy aktuál-ní adresář). Kromě toho by cesta neměla obsahovat adresáře, do nichž je možné volně zapisovat, protože tím by mohl případný útočník modifikovat soubory, které superuživatel spouští.
Jako superuživatel nikdy nepoužívejte nástroje *rlogin/rsh/rexec* (říká se jim r-utility). Jsou cílem celé řady útoků a pro superuživatele jsou obzvlášť nebezpečné. Superuživatel by neměl mít vytvořen soubor .rhosts.
Soubor /etc/security obsahuje seznam terminálů, z nichž se může superuživatel přihlá-sit. Implicitně (RedHat Linux) je povoleno přihlášení pouze z lokálních virtuálních termi-nálů. Při rozšiřování tohoto souboru buďte velmi ostražití. Vždy byste měli být schopni se vzdáleně přihlásit jako normální uživatel a pak teprve přejít do režimu superuživatele (nej-lépe s použitím nějaké bezpečné metody přihlášení, viz kapitolu „ssh a telnet“). Není tedy důvod povolovat přímé přihlášení superuživatele.
Jako superuživatel postupujte vždy pomalu a rozvážně. Vaše akce mohou ovlivnit řadu věcí. Přemýšlejte, než něco napíšete!

Pokud absolutně nezbytně potřebujete někomu poskytnout přístup superuživatele k vašemu systé-mu, existuje několik nástrojů, které vám mohou pomoci. Příkaz sudo umožňuje běžným uživate-lům spouštět některé příkazy používané superuživatelem. Můžete tak například připojovat a odpo-jovat vyjímatelná média, ale už nic víc. Kromě toho příkaz sudo eviduje veškerá svá úspěšná i neú-spěšná použití, takže vidíte, co který uživatel dělal. Z těchto důvodů se dá příkaz sudo použít i tam, kde má superuživatelská práva více lidí, neboť umožňuje sledovat provedené zásahy.

Přestože příkaz sudo umožňuje poskytnout konkrétním uživatelům konkrétní práva ke konkrét-ním operacím, má svá nebezpečí. Měl by se používat pouze pro omezený seznam úkonů, jako jsou například restart serveru nebo vytvoření nového uživatele. Programy, které umožňují vstup do příkazového interpretu, dávají přístup k superuživatelskému účtu každému, kdo je přes sudo spustí. Typicky jde o většinu editorů. Dokonce i tak nevinné programy jako například cat lze pou-žít k přepsání souborů a získat tak privilegia superuživatele. Považujte program sudo spíše za nástroj pro sledování provedených operací, neočekávejte od něj, že nahradí superuživatelský účet a stále bude bezpečný.

Bezpečnost souborů a souborového systému

Několik minut příprav a plánování před aktivací systému vám může pomoci ochránit systém a data.

Neměl by být důvod umožňovat spouštění SUID/SGID programů z uživatelských adresá-řů. Oddíly, na něž může zapisovat i jiný uživatel než root, by měly být v souboru /etc/fstab připojeny s volbou nosuid. Na oddílech s uživatelskými daty můžete také použít volby nodev a noexec, stejně jako na svazku /var, čímž zabráníte spouštění pro-gramů a vytváření znakových a blokových zařízení, která by stejně neměla být k ničemu zapotřebí.
Pokud nabízíte souborové systémy pomocí NFS, nastavte /etc/exports tím nejstriktnějš-ím možným způsobem. Zakažte použití zástupných znaků, zápis jako superuživatel, a je--li to možné, vždy nabídněte přístup pouze pro čtení.

Nastavte umask uživatelů co nejpřísněji, více viz „Nastavení hodnoty umask“.
Pokud připojujete síťové souborové systémy, jako je NFS, nastavte /etc/exports co nej-přísněji. Vhodné jsou typicky volby nodev, nosuid, případně noexec.
Nastavte limity souborového systému, nepoužívejte implicitní nastavení bez limitů. Uživa-telské limity můžete nastavovat pomocí modulu PAM pro řízení prostředků a souboru /etc/pam.d/limits.conf. Například limity pro skupinu uživatelů mohou vypadat takto:

@users hard core 0 @users hard nproc 50 @users hard
rss 5000

Tímto se zakazuje vytvoření výpisu (souborů) core, omezuje se počet procesů na 50 a omezujese využití paměti na 5 MB.Stejná omezení můžete nastavit v konfiguračním souboru /etc/login.defs.

Soubory /var/log/wtmp a /var/run/utmp obsahují záznamy o přihlášení všech uživatelů. Je nutné udržovat jejich integritu, protože z nich lze zjistit, kdy a odkud se uživatel (nebo potenciální útočník) k systému přihlásil. Tyto soubory by měly mít nastavena práva 644.
Pomocí příznaku immutable je možné chránit před neúmyslným přepsáním nebo smazá-ním důležité soubory. Zabraňuje zároveň ve vytvoření pevného odkazu na souboru. Podrobnosti o tomto příznaku naleznete na manuálové stránce příkazu chattr.
SUID a SGID soubory představují potenciální bezpečnostní riziko a je třeba je pečlivě sledovat. Protože tyto programy poskytují speciální práva uživateli, který je spustí, je nutné

> zajistit, aby takto nebyly instalovány nebezpečné programy. Oblíbený trik útočníků je využít programy SUID-root a ponechat si je jako zadní vrátka do systému pro dobu, kdy bude původní díra odstraněna.

Zjistěte si všechny SUID a SGID soubory v systému a pravidelně sledujte, co se s nimi děje, abyste zaznamenali změny provedené případným útočníkem. Všechny SUID/SGID programy naleznete příkazem:

root# find / -type f \(-perm -04000 -o -perm -02000 \)

Distribuce Debian spouští každou noc úlohu, která zjišťuje všechny existující SUID soubory a porovnává je se soubory nalezenými minulou noc. Záznamy o zjištěných výsledcích naleznete ve /var/log/setuid*.

U podezřelých programů můžete příkazem chmod odstranit příznak SUID/SGID a obnovit jej pouze v případě nutnosti.

- Další bezpečnostní díru mohou představovat soubory s právem zápisu, zejména systémo-vé soubory, pokud útočník získá přístup do systému a provede jejich modifikaci. Nebez-pečné jsou i zapisovatelné adresáře, protože útočník může podle libosti vytvářet a mazat soubory. Všechny soubory s globálním právem zápisu najdete příkazem:

  root# find / -perm -2 ! -type l –ls

  Vždy byste měli přesně vědět, proč zrovna tyto soubory mají právo zápisu. Za normálních okolností jsou některé soubory globálně zapisovatelné, například některé soubory v /dev a některé symbolické odkazy, proto parametr ! -type l, kterým jsme je z hledání vyloučili.
- Dalším příznakem přítomnosti útočníka v systému mohou být soubory bez vlastníka. Soubory, které nemají vlastníka nebo nepatří žádné skupině, najdete příkazem:

  root# find / \(-nouser -o -nogroup \) –print

- Pravidelně byste měli vyhledávat soubory .rhosts a nepovolovat jejich existenci. Neza-pomínejte, že útočníkovi pak stačí získat přístup k jedinému účtu a může mít přístup k celé síti. Soubory .rhosts naleznete příkazem:

  root# find /home -name .rhosts –print

- A konečně, než změníte přístupová práva jakéhokoliv systémového souboru, ujistěte se, že přesně rozumíte tomu, co děláte. Nikdy neměňte práva souborů jen proto, že je to jedno-duchá náprava toho, že něco nefunguje. Před změnou práv si vždy zjistěte, proč soubor zrovna takováto práva má.

Nastavení hodnoty umask

Příkazem umask můžete zjistit implicitní práva vytvářených souborů. Jedná se o osmičkový dopl-něk požadovaných práv. Pokud se soubory vytvářejí bez ohledu na přístupová práva, může uži-vatel omylem poskytnout právo ke čtení nebo zápisu někomu, komu nechtěl. Typická nastavení hodnoty umask jsou 022, 027 a 077 (které je nejpřísnější). Normálně se tato hodnota nastavuje v /etc/profile a platí tak pro všechny uživatele systému. Výsledná přístupová práva se vypo-čtou takto: Implicitní práva user/group/others (7 pro adresáře, 6 pro soubory) zkombinujeme po jednotlivých bitech s inverzní maskou (NOT) pomocí AND.

soubor, implicitně 6, binárně: 110, maska 2: 010, NOT: 101
výsledná práva, AND: 100 (tj. 4, r--)

soubor, implicitně 6, binárně: 110, maska 6: 110, NOT: 001
výsledná práva, AND: 000 (tj. 0, ---)

adresář, implicitně 7, binárně: 111 maska. 2: 010, NOT: 101
výsledná práva, AND: 101 (tj. 5, r-x)

adresář, implicitně 7, binárně: 111 maska 6: 110, NOT: 001
výsledná práva, AND: 001 (equals 1, --x)

Uživatel root by měl mít nastavenu masku 077, což vyřadí práva čtení, zápisu a spuštění všem ostatním uživatelům, pokud nebudou práva explicitně změněna příkazem chmod. V tomto pří-padě budou mít nově vytvářené adresáře práva 744, získaná odečtením hodnoty 033 od 777. Nově vytvořené soubory s maskou 033 budou mít práva 644.

Pokud používáte RedHat a jejich mechanismus vytváření uživatelů a skupin, stačí použít hodnotu masky 002. Je to dáno tím, že standardně je v každé skupině pouze jeden uživatel.

Práva souborů

Je důležité zajistit, že systémové soubory nemohou otevřít a upravovat uživatelé a skupiny, kteřínemají správu systému provádět.Unix řídí přístupová práva k souborům a adresářům pro tři kategorie: vlastníka, skupinu a ostatní.Vždy existuje právě jeden vlastník, libovolný počet členů skupiny a pak všichni ostatní.

Rychlé vysvětlení přístupových práv v Unixu:Vlastnictví – který uživatel a skupina mají právo nastavovat přístupová práva i-uzlu a rodičov-ského i-uzlu.

Oprávnění – bitové příznaky, které je možno nastavovat a rušit, čímž se umožňují určité typy pří-stupu.Oprávnění pro adresáře mohou mít jiný význam než stejná oprávnění pro soubory.Čtení:

Právo prohlížet si obsah souboru.
Právo číst adresář.

Zápis:

Právo přidat nebo změnit soubor.
Právo mazat nebo přesouvat soubory v adresáři.

Spuštění:

Právo spustit binární soubor nebo skript.
Právo prohledávat adresář v kombinaci s právem čtení.

Příznak Save Text (pro adresáře)Takzvaný „sticky bit“ má jiný význam pro adresáře a jiný pro soubory. Je-li nastaven u adresáře,může uživatel mazat pouze soubory, jichž je vlastníkem nebo k nimž má explicitně právo zápisu,a to i v případě, že má právo zápisu do adresáře. Tento příznak je určen pro adresáře jako */tmp*,
do nějž sice mohou zapisovat všichni uživatelé, ale není žádoucí, aby kdokoliv mohl cokolivmazat. Nastavení tohoto příznaku se ve výpisu adresáře indikuje symbolem *t*.Příznak SUID (pro soubory)Tento příznak umožňuje souboru provést operaci *set-user-id*. Jestliže je u souboru nastaven pří-
znak SUID a soubor je spustitelný, pak spuštěné procesy budou mít práva vlastníka souborua nikoliv uživatele, který proces spustil. Toto nastavení je zodpovědné za řadu průniků založe-ných na chybách typu přetečení bufferu.

Příznak SGID (pro soubory)Tento příznak povoluje operaci *set-group-id*. Jde o podobné chování jako u příznaku SUID, v tomto případě se však nastavení týká skupiny, pod níž proces běží. Aby měl příznak efekt, musíbýt soubor spustitelný.Příznak SGID (pro adresáře)Jestliže je příznak SGID nastaven pro adresář (příkazem chmod g+s adresář), soubory vytváře-
né v adresáři budou mít skupinu stejnou, jako je skupina adresáře. Vy – vlastník souboru.Skupina – skupina, do níž patříte.Kdokoliv – kdokoliv v systému, kdo není vlastníkem ani členem skupiny.Příklad souboru:

-rw-r--r-- 1 kevin users 114 Aug 28 1997 .zlogin
bit – adresář? (ne)

bit – čtení vlastníkem? (ano, kevinem)
bit – zápis vlastníkem? (ano, kevinem)
bit – spuštění vlastníkem? (ne)
bit – čtení skupinou? (ano, users)
bit – zápis skupinou? (ne)
bit – spuštění skupinou? (ne)
bit – čtení kýmkoliv? (ano)
bit – zápis kýmkoliv? (ne)
bit – spuštění kýmkoliv? (ne)

Následující řádky představují příklady minimálních přístupových práv, která jsou potřebná k pro-vedení popsané akce. Můžete samozřejmě nastavit větší oprávnění, tento přehled ukazuje, jaká mají jednotlivá oprávnění efekt:

-r-------- Umožňuje čtení vlastníkem .
--w------- Umožňuje vlastníkovi soubor změnit nebo smazat. (Kdokoliv, kdo má právo zápisu pro adresář, v němž se soubor nachází, jej může přepsat a tedy smazat.)
---x------ Vlastník může soubor spustit, ne však, jde-li o skript, u nějž je nutné i právo čtení. ---s------ Soubor bude spuštěn s efektivním uživatelským ID odpovídajícím vlastníkovi. --------s- Soubor bude spuštěn s efektivním skupinovým ID odpovídajícím skupině. -rw------T Neprovádět aktualizaci posledního času modifikace. Používá se typicky pro swapovací soubory. ---t------ Nemá význam (dříve tzv. sticky bit).

Příklad adresáře:

drwxr-xr-x 3 kevin users 512 Sep 19 13:47 .public_html/
bit – adresář? (ano, obsahuje spoustu souborů)
bit – čtení vlastníkem? (ano, kevinem)
bit – zápis vlastníkem? (ano, kevinem)
bit – prohledávání vlastníkem? (ano, kevinem)
bit – čtení skupinou? (ano, users)
bit – zápis skupinou? (ne)
bit – prohledávání skupinou? (ano, users)
bit – čtení kýmkoliv? (ano, kýmkoliv)
bit – zápis kýmkoliv? (ne)
bit – prohledávání kýmkoliv? (ano, kýmkoliv)

Následující řádky představují příklady minimálních přístupových práv, která jsou potřebná k pro-vedení popsané akce. Můžete samozřejmě nastavit větší oprávnění; tento přehled ukazuje, jaká mají jednotlivá oprávnění efekt:

dr--------Obsah je možné vypsat, ale nelze číst atributy souborů.d--x------Do adresáře je možno vstoupit a lze jej použít ve spouštěcí cestě.dr-x------Vlastník může číst atributy souborů.d-wx------Soubory je možné vytvářet/mazat, i když adresář není nastaven jako aktuální. d------x-t Zabraňuje ve smazání souborů někým jiným než vlastníkem. Používá se u adresáře/tmp. d---s---s--- Bez efektu.

Systémové konfigurační soubory (obvykle v adresáři /etc) mají typicky práva 640 (-rw-r-----) a vlastní je root. V závislosti na požadované míře zabezpečení to můžete změnit. Žádné systémo-vé soubory by neměly být zapisovatelné skupinou a kýmkoliv. Některé konfigurační soubory, například /etc/shadow, může číst jenom root. Adresáře v /etc by minimálně neměly být pří-stupné komukoliv.

SUID skripty

SUID skripty představují závažné bezpečnostní riziko, a proto jádro v tomto případě dané nasta-vení nerespektuje. Bez ohledu na to, za jak bezpečný skript považujete, útočník jej může využít k získání superuživatelského příkazového interpretu.

Kontrola integrity

Další vhodný způsob detekce lokálních i síťových útoků je použití nějakého systému pro kontro-lu integrity, jako je například Tripwire, Aide nebo Osiris. Tyto programy vypočtou řadu kontrol-ních součtů důležitých binárních a konfiguračních souborů a kontrolují je proti databázi původ-ních, správných hodnot. Tím se prozradí jakékoliv změny v souborech. Rozumné je tento typ pro-gramů nainstalovat na disketu a pak na disketě fyzicky zakázat zápis.

Tímto způsobem zamezíte útočníkovi modifikovat samotný kontrolní program nebo jeho databázi. Jakmile něco podobného nastavíte, je rozumné kontrolu pravidelně spouštět, abyste poznali, zda nedošlo ke změně.

Můžete například přidat záznam do tabulky démona cron, kterým spustíte kontrolu každou noc, a výsledek si necháte odeslat e-mailem. Nastavení typu:

```
# nastavení adresáta MAILTO=kevin # spuštění Tripwire 15 05 * * * root /usr/local/adm/tcheck/tripwire
```

provede kontrolu a pošle zprávu každý den v 5.15 ráno.

Kontrola integrity je vynikající nástroj pro detekci útoků dříve, než byste si jich všimli jinak. Na

normálním systému se ovšem řada souborů mění, takže musíte pečlivě dávat pozor na to, co jsounormální změny a co může být projevem útoku.Program Tripwire můžete zdarma získat na adrese *http://www.tripwire.org.* Můžete si doplatit za

manuály a podporu.Aide najdete na adrese *http://www.cs.tut.fi/~rammer/aide.html.*Osiris najdete na adrese *http://www.shmoo.com/osiris/.*

Trojské koně

Trojské koně jsou pojmenovány po bájné léčce z Homérovy Iliady. Myšlenka je taková, že útoč-ník distribuuje program, který vypadá skvěle, a přesvědčuje uživatele, aby si program nahráli a spustili jako root. Pak program nepozorovaně naruší bezpečnost systému. Zatímco si tedy uži-vatelé myslí, že program dělá nějakou jednu věc (kterou může klidně dělat velmi dobře), před-stavuje současně i útok na systém.

Měli byste si dávat pozor na to, jaké programy na systém instalujete. RedHat používá ve svých souborech RPM kontrolní součty MD5 a GPG podpisy, takže si můžete ověřit, zda opravdu insta-lujete originál. Ostatní distribuce používají podobné metody. Nikdy nespouštějte žádné neznámé binární programy, od nichž nemáte zdrojové kódy. Útočníci obvykle nedistribuují své programy společně se zdrojovými kódy, aby je mohl kdokoliv prozkoumat.

I když to může být komplikované, vždy si ověřte, zda zdrojový kód programu opravdu pochází z distribučního serveru programu. Pokud budete program spouštět jako superuživatel, vždy si pro-hlédněte a zkontrolujte jeho zdrojový kód, ať už sami nebo požádejte někoho důvěryhodného.

Hesla a šifrování

Jedním z nejvýznamnějších prvků k zajištění bezpečnosti jsou dnes hesla. Pro vás i vaše uživate-le je důležité používat bezpečná, neuhodnutelná hesla. Většina moderních linuxových distribucí obsahuje program passwd, který vám nedovolí nastavit snadno uhodnutelné heslo. Zkontrolujte, zda program passwd ve vašem systému je aktuální a obsahuje tuto funkci.

Hlubší debata o šifrování je mimo záběr tohoto dokumentu, nicméně jistý úvod je na místě. Šif-rování je velice užitečné, dnes možná nezbytné. Existuje velké množství šifrovacích metod, každá má své význačné vlastnosti.

Většina Unixů (Linux nevyjímaje) primárně používá k uložení hesel jednosměrné šifrování, založe-né na algoritmu DES (Data Encryption Standard). Takto zašifrovaná hesla se pak typicky ukládají v souboru /etc/passwd nebo (méně typicky) v souboru /etc/shadow. Když se přihlašujete, pro-vede se nové zašifrování zadaného hesla a to se porovná s údajem v souboru, kde jsou hesla ulo-žena. Pokud se výsledky shodují, musí jít o stejné heslo a povolí se přihlášení do systému. I když DES je sám o sobě obousměrný algoritmus (umožňuje data zašifrovat a pak i dešifrovat), modifi-kace používaná ve většině Unixů je jednosměrná. Znamená to, že by nemělo být možné obrátit směr šifrování a ze souboru /etc/passwd (nebo /etc/shadow) zjistit původní podobu hesla.

Útoky hrubou silou (například programy Crack nebo John the Ripper, viz část „Crack a John the Ripper“) často umožní uhodnout heslo, pokud nebylo zvoleno dostatečně náhodně. Moduly PAM (viz níže) umožňují šifrovat hesla jinými algoritmy (například MD5 a podobně). I zde můžete použít program Crack. Doporučujeme tento program pravidelně spouštět proti vaší vlastní databázi
hesel a hledat tak slabá hesla. Pak kontaktujte příslušného uživatele a doporučte mu změnu hesla. Návod na volbu dobrých hesel naleznete na adrese *http://consult.cern.ch/writeup/security/ security_3.html.*

PGP a kryptografie s veřejným klíčem

Kryptografie s veřejným klíčem, například algoritmy používané programem PGP, používá jeden klíč pro zašifrování a jiný pro rozšifrování. Klasická kryptografie naproti tomu používá stejný klíč pro zašifrování i rozšifrování. Tento klíč musí znát obě komunikující strany, a je tedy nutné nějakým bezpečným způsobem zajistit jeho přenos od jedné strany ke druhé.

Aby se vyloučila potřeba bezpečného přenosu šifrovacího klíče, používá kryptografie s veřejným klíčem dva klíče: veřejný klíč a privátní klíč. Veřejný klíč každého je všem k dispozici pro účely zašifrování, privátní klíč udržuje každý účastník v tajnosti a používá jej k rozšifrování zpráv zašifrovaných příslušným veřejným klíčem.

Klasická kryptografie i kryptografie s veřejným klíčem mají své výhody a o jejich vlastnostech se můžete dočíst v dokumentu *RSA Cryptography FAQ*, zmíněném i na konci této kapitoly. Algoritmus PGP (Pretty Good Privacy) je v Linuxu široce podporován. Verze 2.6.2 a 5.0 jsou pova-žovány za správně fungující. Dobrý úvod do PGP a návod k jeho použití naleznete v dokumentu *PGP FAQ* na adrese *http://www.pgp.com/service/export/faq/55faq.cgi.*

Ujistěte se, že používáte verzi vhodnou pro vaši zemi. Vzhledem k exportním omezením americké vlády se nepovoluje elektronický přenos silných šifrovacích algoritmů mimo území USA.

Exportní pravidla v USA nyní určuje EAR (Export Administration Regulations), nikoliv ITAR. Podrobný návod ke konfiguraci PGP v Linuxu naleznete na adrese *http://mercury.chem. pitt.edu/~angel/LinuxFocus/English/November1997/article7.html.* Tento návod byl psán pro mezinárodní verzi PGP, lze jej však použít i pro americkou verzi. Pro některé z nejnovějších verzí Linuxu budete možná potřebovat některé doplňky, které najdete na adrese ftp:// metalab.unc.edu/pub/Linux/apps/crypto.

Existuje projekt na reimplementaci PGP s otevřeným kódem. GnuPG představuje úplnou a zdar-ma dostupnou náhradu programu PGP. Protože nepoužívá algoritmy IDEA ani RSA, lze jej použít bez omezení. GnuPG odpovídá standardu OpenPGP. Další informace naleznete na stránkách *GNU Privacy Guard* na adrese *http://www.gnupg.org/.*

Řada informací o kryptografii je v již zmíněném dokumentu *RSA Cryptography FAQ*, který může-te najít na adrese *http://www.rsa.com/rsalabs/newfaq/.* Naleznete v něm informace o pojmech, jako jsou „Diffie-Hellmanův algoritmus“, „kryptografie s veřejným klíčem“, „digitální certifikáty“ a podobně.

SSL, S-HTTP a S/MIME

Uživatelé se často ptají na rozdíly mezi různými zabezpečovacími a šifrovacími protokoly a na to, jak je použít. I když nečtete text o šifrování, bude rozumné si jednotlivé protokoly stručně před-stavit a ukázat, kde nalézt další informace.

- SSL, Secure Sockets Layer, je šifrovací metoda vyvinutá společností Netscape k zajištění bezpečné komunikace po Internetu. Podporuje několik různých šifrovacích protokolů a poskytuje autentizaci klientů a serverů. SSL funguje na transportní vrstvě, vytváří bez-pečný šifrovaný datový kanál, a může tak transparentně šifrovat různé typy dat. Nejčastěji se s touto metodou potkáme při návštěvě zabezpečených stránek a prohlížení zabezpeče-ných dokumentů, kde zajišťuje bezpečnou komunikaci mezi prohlížečem a serverem. Další informace můžete najít na adrese *http://www.consensus.com/security/ssl-talk-faq.html.*
  Informace o dalších bezpečných protokolech společnosti Netscape naleznete na adrese *http://home.netscape.com/info/security-doc.html.* Stojí také za zmínku, že protokol SSL lze použít k přenosu řady jiných běžných protokolů, k jejich bezpečnému „obalení“. Viz *http://www.quiltaholic.com/rickk/sslwrap/.*

S-HTTP je další protokol zajišťující bezpečnostní služby na Internetu. Byl navržen k zajiš-tění utajení, autentizace, integrity a nepopíratelnosti. V tomto kontextu znamená: autenti-zace = jednoznačné prokázání totožnosti, integrita = nemožnost narušení (modifikace) pře-nášených dat třetí stranou, nepopíratelnost = nemožnost popřít zprávu, jejímž jsem auto-rem. Podporuje více mechanismů správy klíčů a šifrovacích algoritmů na základě dohody mezi účastníky každé transakce. S-HTTP je omezen na specifické programy, které jej imple-mentují, a každou zprávu šifruje samostatně.
S/MIME, Secure Multipurpose Internet Mail Extension, je šifrovací standard používaný k šif-rování elektronické pošty a dalších typů zpráv na Internetu. Jedná se o otevřený standard vyvíjený společností RSA, takže se dá očekávat, že v brzké době bude jeho implementace i v Linuxu. Podrobnější informace o protokolu S/MIME můžete najít na adrese *http://home. netscape.com/assist/security/smime/overview.html.*

Implementace IPSEC v Linuxu

Kromě CIPE a dalších metod šifrování dat existuje v Linuxu i několik implementací IPSEC. IPSECje standard navržený IETF, který vytváří kryptograficky zabezpečenou komunikaci na síťové vrst-vě IP a zajišťuje autentikaci, integritu, řízení přístupu a utajení. Informace o IPSEC naleznete naadrese *http://www.ietf.org/html.charters/ipsec-charter.html.* Tam najdete také odkazy na další pro-tokoly zajišťující správu klíčů, poštovní konferenci o IPSEC a archivy.

Linuxová implementace *x-kernel*, vyvíjená na arizonské univerzitě, používá k implementaci síťo-vých protokolů objektovou základnu nazvanou x-kernel. Naleznete ji na adrese*http://www.cs.arizona.edu/xkernel/hpcc-blue/linux.html.* Jednoduše řečeno, *x-kernel* je metodapředávání zpráv na úrovni jádra, což usnadňuje další implementaci.

Další zdarma dostupná linuxová implementace IPSEC je Linux FreeS/WAN IPSEC. Na webovýchstránkách tohoto projektu se dočtete, že: „Tato služba umožňuje vytvářet bezpečné tunely přesnezabezpečené sítě. Cokoliv přenášeného po nezajištěné síti je šifrováno branou IPSEC a dešifru-je se branou na druhém konci tunelu. Výsledkem je virtuální privátní síť, VPN. Jedná se o síť, kteráje „privátní“, přestože zahrnuje počítače a systémy na „neprivátním“ Internetu.“

Tento program naleznete na adrese *http://www.xs4all.nl/~freeswan/* a v době vzniku tohoto textu
se nachází ve verzi 1.0.Stejně jako u jiných kryptografických produktů není distribuován jako součást jádra vzhledemk platným exportním omezením.

ssh (Secure Shell) a stelnet

Secure Shell (ssh) a stelnet jsou z rodiny programů, které umožňují připojení ke vzdáleným počítačům zašifrovaným kanálem.Openssh je rodina programů, které slouží jako bezpečná náhrada programů rlogin, rsh a rcp.K zašifrování komunikace mezi účastníky a k autentizaci uživatelů používá kryptografii s veřejnýmklíčem. Umožňuje bezpečné přihlášení ke vzdálenému počítači nebo kopírování dat mezi počíta-či s vyloučením útoku typu „man-in-the-middle“ a s

vyloučením falšování záznamů DNS. Zajišťu-je rovněž kompresi dat a zabezpečenou X Window komunikaci.
V současnosti existuje několik implementací ssh. Původní komerční implementaci společnosti

Data Fellows naleznete na adrese *http://www.datafellows.com.*Vynikající implementace Openssh je založena na původní verzi ssh společnosti Data Fellows a jeúplně přepracována, takže neobsahuje žádné patentované nebo proprietární části. Je k dispozicizdarma pod licencí BSD. Naleznete ji na adrese *http://www.openssh.com*.
Dále existuje projekt úplné reimplementace ssh od začátku, pojmenovaný „psst…“. Naleznete jej

na adrese *http://www.net.lut.ac.uk/psst/*.Program ssh můžete použít k připojení k linuxovému serveru i ze stanic s Windows. Existuje něko-lik zdarma dostupných implementací klienta pro Windows, například *http://guardian.htu.tuwien.ac.at/ therapy/ssh/*, a také komerční implementace od Data Fellows na adrese*http://www.datafellows.com.*

SSLeay je volně šiřitelná implementace protokolu Netscape's Secure Sockets Layer, kterou vyvinulEric Young. Obsahuje několik aplikací, jako např. Secure telnet, modul pro program Apache,několik databází a také několik algoritmů, mimo jiné např. DES, IDEA a Blowfish.

Pomocí této knihovny bylo vytvořeno bezpečné nahrazení programu telnet, které šifruje přes tel-netové spojení. Na rozdíl od SSH stelnet používá SSL, což je protokol Secure Sockets Layer vyvi-nutý firmou Netscape. Se Secure telnet a Secure FTP je nejlépe začít od často kladených otázek(FAQ) na adrese *http://www.psy.uq.oz.au/~ftp/Crypto/.*

SRP je další bezpečná implementace protokolů telnet a ftp. Z jejich webových stránek uvádíme:„Projekt SRP vyvíjí bezpečné internetové programy k použití zdarma. Počínaje plně zabezpečený-
mi distribucemi protokolů Telnet a FTP doufáme, že nahradíme slabé síťové autentizační systémysilnými bez újmy na snadnosti použití. Bezpečnost by měla být standard, ne volba!“ Další informace najdete na adrese *http://www-cs-students.stanford.edu/~tjw/srp/.*

PAM – Pluggable Authentication Modules

Nové distribuce systémů RedHat a Debian se dodávají s unifikovaným autentizačním mechanis-mem, nazvaným „PAM“. PAM dovoluje změnit používané autentizační metody a požadavky za běhu a zapouzdření všech lokálních autentizačních metod bez nutnosti znovu překládat jednotli-vé programy. Konfigurace mechanismu PAM je mimo rámec tohoto dokumentu. Podrobnosti se můžete dozvědět na webových stránkách PAM na adrese *http://www.kernel.org/pub/linux/ libs/pam/index.html.*

Uveďme si jen několik věcí, které je možné pomocí PAM dosáhnout:

Šifrování hesel jiným algoritmem než DES. (Čímž se komplikuje jejich odhalení útokem hrubou silou.)
Nastavení limitů na jednotlivé prostředky, takže nelze provést útoky typu DoS. (Jde o limi-ty na počty procesů, spotřebu paměti a podobně.)
Jednoduchá aktivace stínových hesel (viz dále).
Povolit jednotlivým uživatelům přihlášení jen v určitý čas a z určitého místa.

Po několika hodinách instalace a konfigurace můžete zabránit spouště útoků ještě dříve, než k nim dojde. Například následujícími řádky v souboru /etc/pam.d/rlogin můžete všem uživatelům zakázat použití souboru .rhosts:

```
# Zákaz rsh/rlogin/rexec login auth required pam_rhosts_auth.so no_rhosts
```

Cryptographic IP Encapsulation (CIPE)

Primárním smyslem tohoto programu je poskytnout možnost vytvoření bezpečného (ve smyslu odolného proti odposlechu, analýze přenosu a podstrčení falešných dat) propojení sítí prostřed nictvím nezabezpečených sítí, jako je Internet.

CIPE šifruje data na síťové úrovni. Pakety cestující mezi počítači v síti jsou šifrovány. Šifrovací

modul je umístěn vedle ovladače, který odesílá a přijímá pakety.
To je rozdíl oproti SSH, které šifruje data nad spojením, na úrovni síťového soketu. Šifruje se logic ké spojení mezi programy běžícími na různých počítačích.

CIPE je možno použít při tunelování, k vytvoření virtuální privátní sítě. Šifrování na nízké úrovnimá výhodu v tom, že může mezi dvěma počítači ve virtuální privátní síti fungovat transparentně,bez nutnosti zásahů do aplikačních programů.
Citujeme z dokumentace k CIPE:„Standard IPSEC definuje skupinu protokolů, které je možno použít (mimo jiné) i k vytváření šif-rovaných virtuálních privátních sítí. Nicméně IPSEC je poměrně těžkopádná a komplikovaná sku-pina protokolů s řadou možností a úplná implementace této skupiny je stále poměrně zřídka pou-živaná a některé problémy (například správa klíčů) stále

nejsou plně vyřešeny. CIPE volí jedno-dušší přístup, kdy se většina volitelných věcí (například šifrovací algoritmus) pevně stanovuje při
instalaci. Tím se sice omezuje flexibilita, ale vzniká jednodušší (a tedy efektivní a snáze testova-telná) implementace.“ Další informace můžete najít na adrese *http://www.inka.de/~bigred/devel/cipe.html.* Stejně jako u jiných kryptografických produktů ani tento není součástí jádra kvůli exportním ome-
zením.

Kerberos

Kerberos je autentikační systém vyvinutý v projektu Athena na MIT. Když se uživatel přihlásí, Ker-beros jej autentikuje (pomocí hesla) a pak uživateli umožní prokázat svou totožnost dalším ser-verům a systémům v síti.

Tato autentikace se používá v programech jako rlogin, které umožňují uživateli přihlašovat se k dalším počítačům bez hesla (a bez použití souboru .rhosts). Tuto metodu lze použít i k zajiš-tění, aby pošta došla správnému příjemci, a k zajištění, že odesilatel pošty je opravdu ten, za koho se vydává.

Kerberos a další programy této skupiny zabraňují uživateli podvést systém a vydávat se za něko-ho jiného. Bohužel, instalace systému Kerberos je velmi náročná, vyžaduje modifikace a náhrady řady standardních programů.

Další informace o systému Kerberos naleznete v dokumentu *Kerberos FAQ* na adrese *http://nii.isi.edu/info/kerberos/.*

Případně: Jennifer G. Stein, Clifford Neuman a Jeffrey L. Schiller. „Kerberos: An Authentication Ser
vice for Open Network Systems.“ USENIX Conference Proceedings, Dallas, Texas, zima 1998. Systém Kerberos by rozhodně neměl být prvním krokem ve zvyšování bezpečnosti systému. Jeho instalace je velmi náročná a není zdaleka tak rozšířený jako například SSH.

Stínová hesla

Stínová (*shadow*) hesla představují mechanismus, jak informace o zašifrovaných heslech udržet skry-té před běžnými uživateli. Nové verze distribucí RedHat a Debian používají stínová hesla implicitně, na jiných systémech jsou však hesla uložena v souboru /etc/passwd, kde je mohou uživatelé volně číst. Kdokoliv pak může spustit nějaký program pro luštění hesel a snažit se je uhodnout. Stínová hesla jsou naproti tomu uložena v souboru /etc/shadow, který je normálním uživatelům nepří-stupný. Abyste mohli stínová hesla použít, musíte zajistit u všech nástrojů, které hesla používají, jejich přeložení s podporou stínových hesel. Výše zmíněný systém PAM umožňuje jednoduché vložení stí-nového modulu, překlad binárních souborů není nutný. Podrobnější informace můžete najít v doku-mentu *Shadow-Password HOWTO* na adrese *http://metalab.unc.edu/LDP/HOWTO/Shadow-Password-HOWTO.html.* Tento dokument je však už poměrně starý a u distribucí používajících PAM není nutný.

„Crack“ a „John the Ripper“

Pokud váš program passwd z nějakého důvodu nevynucuje zadání obtížně uhodnutelných hesel, můžete vyzkoušet nějaký program pro luštění hesel a ověřit si, zda vaši uživatelé používají bez-pečná hesla.

Programy pro luštění hesel jsou založeny na jednoduchém principu: Zkoušejí jednotlivá slova a jejich variace ze slovníku, každé zašifrují a porovnají s uloženým heslem. Pokud dojde ke shodě, podařilo se jim uhodnout heslo.

Existuje celá řada programů tohoto typu, nejznámější jsou Crack a John the Ripper (*http://www.openwall.com/john/*). Jejich běh zabere hodně procesorového času, můžete si však ověřit, zda by je útočník nemohl využít, a to tak, že je nejprve vyzkoušíte sami a uvědomíte uži-vatele se slabými hesly. Útočník sice bude potřebovat jinou bezpečnostní díru k přečtení soubo-ru /etc/passwd, ale takové díry jsou běžnější, než byste si mysleli.

Protože bezpečnost je dobrá jen tak, jako nejslabší počítač v síti, stojí za zmínku, že pokud v síti máte počítače s Windows, měli byste vyzkoušet program L0pthCrack, implementaci programu Crack pro Windows. Najdete ji na adrese *http://www.l0pht.com.*

CFS – Cryptographic File System a TCFS – Transparent Cryptographic File System

CFS je metoda umožňující šifrovat celé adresářové stromy a umožňující uživatelům ukládat zašif-rované soubory. Využívá služeb NFS serveru na lokálním počítači. RPM balíčky jsou dostupné na adrese *http://www.zedz.net/redhat/*, podrobnější informace o fungování systému pak na adrese ftp://ftp.research.att.com/dist/mab/.

TCFS je vylepšení CFS o lepší integraci se souborovým systémem, takže celý mechanismus šifro
vání je pro uživatele transparentní. Další informace naleznete na adrese *http://www.tcfs.it/.* Tento systém rovněž nemusíte použít na celý souborový systém, funguje i na část adresářového stromu.
Další možnosti, jak používat šifrovaný souborový systém, naleznete na *http://www.kerneli.org/.*

X Window, SVGA a bezpečnost displeje X Window

Je důležité zabezpečit grafický displej, aby se útočníkům zabránilo zmocnit se hesel, která zapisujete, číst dokumenty nebo informace, jež si čtete na obrazovce, nebo dokonce využít nějaké díry k získání práv superuživatele. U vzdáleného spouštění grafických aplikací přes síť rovněž existuje riziko odposlechu, kdy útočník může zaznamenat veškerou vaši interakci se vzdáleným systémem.

Systém X Window obsahuje řadu mechanismů pro řízení přístupu. Jedním z nejjednodušších je mechanismus založený na počítači. Pomocí xhost můžete specifikovat, ze kterých počítačů je možný přístup k vašemu displeji. To ovšem není příliš bezpečné, protože jakmile někdo získá pří-stup k vašemu počítači, může doplnit i svůj počítač a snadno se tak dostat dovnitř. Pokud navíc
povolujete přístup z nedůvěryhodných počítačů, kdokoliv na nich může získat přístup k vašemu

displeji. Pokud používáte k přihlášení *xdm* (X Display Manager), máte možnost použít bezpečnější přístu-povou metodu: MIT-MAGIC-COOKIE-1. Ve vašem souboru .Xauthority bude uložen 128bitovývygenerovaný „cookie“. Pokud potřebujete povolit přístup k vašemu displeji vzdálenému počíta-či, můžete použít příkaz *xauth* a informace v souboru *.Xauthority* a povolit přístup pouze pro totojedno připojení, viz dokument *Remote-X-Apps mini-howto* na adrese *http://metalab.unc.edu/LDP/HOWTO/mini/Remote-X-Apps.html.*

Můžete také použít *ssh* (viz kapitolu „ssh a telnet“) a provozovat zabezpečená X spojení. Výhodaje také v tom, že mechanismus je pro koncového uživatele transparentní a po síti se nepřenášejížádná nešifrovaná data.

Jakákoliv vzdálená připojení k X serveru můžete zakázat parametrem -nolisten tcp X serveru. Tím
se zabrání jakýmkoliv síťovým spojením na server.Podívejte se na manuálovou stránku Xsecurity, kde naleznete více informací o zabezpečení syste-mu X Window. Nejbezpečnější řešení je použít *xdm* k přihlášení se na vaši konzolu a pak sshk přihlášení na vzdálené systémy, kde chcete spouštět grafické aplikace.

SVGA

Programy SVGAlib jsou typicky SUID-root, aby mohly přímo přistupovat k videozařízení počítače. Tím jsou velmi nebezpečné. Pokud dojde k jejich havárii, typicky budete muset restartovat počí-tač. Ujistěte se, že všechny spouštěné SVGA programy jsou autentické a důvěryhodné. Ještě lepší je vůbec je nespouštět.

GGI (Generic Graphics Interface project)

Projekt GGI se snaží řešit některé problémy s videorozhraními v Linuxu. GGI přesouvá některé malé části videokódu přímo do jádra a pak řídí přístup k videosystému. Znamená to, že GGI může v kterémkoliv okamžiku obnovit konzolu do známého dobrého stavu. Také nabízí bezpečnostní mechanismy zabraňující ve spuštění trojských koňů, snažících se odposlechnout heslo. Více viz *http://synergy.caltech.edu/~ggi/.*

Bezpečnost jádra

V této části kapitoly uvádíme popis voleb při konfiguraci jádra, které se vztahují k bezpečnosti,
vysvětlení jejich funkcí a způsobu použití. Vzhledem k tomu, že připojení počítače k síti je řízeno jádrem, je důležité zajistit vysokou míru jeho bezpečnosti bez jakýchkoli kompromisů. Abyste předešli útokům po síti, je nutno průběžně používat aktuální verzi jádra. Nová jádra naleznete na adrese ftp://ftp.kernel.org nebo u svého prodejce.

Existuje také mezinárodní skupina, která poskytuje jednotnou kryptografickou záplatu do klasic-kého linuxového jádra. Tato záplata podporuje řadu kryptografických subsystémů a dalších věcí, které nemohou být součástí jádra z důvodů vývozních omezení. Více informací naleznete na jejich internetové stránce: *http://www.kerneli.org.*

Volby při překladu jádra 2.0

Pro jádro 2.0 platí následující volby. Měli byste se s nimi seznámit při konfiguraci jádra. Většina komentářů je z./linux/Documentation/Configure.help, což je stejný dokument, z kterého čerpá nápověda ve fázi make config při překladu jádra.

- Network Firewalls (CONFIG_FIREWALL) Tato volba by měla být aktivní, pokud na linuxovém počítači hodláte provozovat firewal
ly nebo maškarádu (překlad adres). Jde-li o běžný klientský počítač, bez nebezpečí lze odpovědět „no“.
IP: forwarding/gatewaying (CONFIG_IP_FORWARD) Pokud aktivujete předávání IP adres, stane ze z Linuxu směrovač. Je-li počítač připojen k síti, můžete předávat data z jedné sítě do druhé a pravděpodobně zničíte firewall, který sem byl dán právě proto, aby k tomu nedocházelo. Běžní uživatelé připojení přes vytáče-né spojení budou chtít tento stav zablokovat a ostatní se budou soustředit na následky,

    které to způsobí. Počítače, které slouží jako firewall, budou chtít tento stav aktivovat a soft
    warově využívat.

Předávání IP adres lze aktivovat dynamicky pomocí příkazu:

```
root# echo 1 > /proc/sys/net/ipv4/ip_forward
```

a zablokovat pomocí příkazu:

```
root# echo 0 > /proc/sys/net/ipv4/ip_forward
```

Měli byste si uvědomit, že soubory v /proc jsou „virtuální“ a jejich velikost nemusí odpo-vídat množství výstupních dat.

- IP: syn cookies (CONFIG_SYN_COOKIES) Útok „SYN“ je útokem způsobujícím odmítnutí služby (*denial of service*, DoS), který spo-třebovává všechny prostředky počítače a vede k nucenému restartu počítače. Důvod k aktivaci si za normálních okolností neumíme představit. V jádru řady 2.2.x povoluje tato konfigurační volba syn cookies, avšak neaktivuje je. Aktivaci je nutno provést pomocí:

```
root# echo 1 > /proc/sys/net/ipv4/tcp_syncookies <P>
```

- IP: Firewalling (CONFIG_IP_FIREWALL) Tato volba je nezbytná, když chcete svůj počítač zkonfigurovat jako firewall nebo chránit pracovní stanice s vytáčeným spojením před napadením z vytáčeného dvoubodového rozhraní.
- IP: firewall packet logging (CONFIG_IP_FIREWALL_VERBOSE) Tato volba zajišťuje poskytování informací o paketech, které dostává firewall. Jde o pakety typu odesilatel, příjemce, port atd.
- IP: Drop source routed frames (CONFIG_IP_NOSR) Tato volba by měla být aktivovaná. Pakety s adresou nastavenou odesilatelem obsahují popis celé cesty k cíli. To znamená, že směrovače, jimiž takové pakety procházejí, je nezkoumají a pouze je pošlou dál. Tak se mohou do systému dostat data, jež představují potenciální riziko.
- IP: masquerading (CONFIG_IP_MASQUERADE) Pokud by některý z počítačů v lokální síti, jemuž Linux tvoří firewall, chtěl odeslat něco ven, tento firewall jej bude „maskovat“, tj. pošle zprávu na cílovou adresu, avšak zpráva bude vypadat, jako by ji odeslal tento firewall. Další podrobnosti viz *http://www. indyramp.com/masq*.

IP: ICMP masquerading (CONFIG_IP_MASQUERADE_ICMP) Tato volba doplňuje předchozí volbu, která maskuje pouze TCP a UDP, o maskování ICMP.

- IP: transparent proxy support (CONFIG_IP_TRANSPARENT_PROXY) Touto volbou přesměrujete firewall tak, že zprávy přicházející z lokální sítě, které jsou určeny pro vzdálený počítač, půjdou na lokální server, který se nazývá „transparentní proxy server“. Lokální počítače se tak domnívají, že komunikují se vzdáleným počítačem, ačkoli jsou připojeny k lokálnímu proxy serveru. Další informace viz Návod k maskování IP adresy na *http://www.indyramp.com/masq.*
- IP: always defragment (CONFIG_IP_ALWAYS_DEFRAG) Obecně je tato volba zablokovaná. Když však vytváříte firewall nebo maškarádu, volbu aktivujete. Data se z jednoho počítače na druhý neposílají jako jediný paket, nýbrž se roz-dělí na několik částí. Problém s touto volbou je ten, že číslo portu obsahuje pouze první fragment. To znamená, že někdo může mezi zbývající pakety vsunout informace, které tam nepatří. Lze tak zabránit útoku typu *teardrop attack* na interní počítač, který ještě nemá pří-slušnou záplatu.
- Packet Signatures (CONFIG_NCPFS_PACKET_SIGNING) Tato volba je k dispozici v jádru řady 2.2.x, která označuje NCP palety (pro lepší zabez-pečení). Běžně se nepoužívá, pro případ potřeby je však k dispozici.

IP: Firewall packet netlink device (CONFIG_IP_FIREWALL_NETLINK) To je skutečně elegantní volba, jejímž prostřednictvím můžete analyzovat prvních 128 bajtů paketu uživatelským programem. Slouží k rozhodování, zda paket v závislosti na jeho platnosti přijmete či nikoli.

Volby při překladu jádra 2.2

V jádru řady 2.2.x je mnoho voleb stejných, vznikly však některé nové. Uvedené popisy jsou pře-vzaty z ./linux/Documentation/Configure.help, což je stejný dokument, z něhož čerpá nápo-věda ve fázi make config při překladu jádra. Uvádíme pouze nově přidané volby, ostatní nalez-nete v popisu jádra 2.0. Nejdůležitější změna v jádru 2.2 je ve firewallu. Místo programu ipfwadm, který zajišťoval firewall ve starší řadě, se nyní používá ipchains.

- Socket Filtering (CONFIG_FILTER) Pro většinu uživatelů je bezpečné odpovědět na tuto volbu „no“. Umožňuje napojit na soc-kety uživatelské filtry a určit, zda mají být pakety přijaty nebo odmítnuty. Pokud nejste nuceni filtrovat provoz a filtr si neumíte sami naprogramovat, neměli byste tuto volbu vyu-žívat. Také je třeba si uvědomit, že tato volba podporuje všechny protokoly kromě TCP. Další informace viz .../linux/Documentation/networking/filter.txt.

Port Forwarding Přenos prostřednictvím portů je dodatek k maskování IP adresy. Pomocí této volby může-me na daných portech přenášet pakety z vnější strany firewallu dovnitř. To můžete použít například k provozování internetového serveru za firewallem nebo k maskování počítače, které mají být přístupné zvnějšku. Externí klient pošle požadavek na firewallový port 80, firewall pošle tento požadavek na internetový server, který jej zpracuje a výsledek pošle přes firewall původnímu klientovi. Klient má dojem, že internetový server provozuje počí-

tač, na němž je firewall. Této volby lze využít i k vyvážení zátěže mezi několik serverů zafirewallem.Informace o této funkci jsou převzaty z *http://www.monmouth.demon.co.uk/ipsubs/portforwarding.html* (nalézt ji můžete pomocí počítače připojeného k Internetu, který je vybaven programy jako např. lynx nebo Netscape). Obecné informace viz ftp://ftp.com-psoc.net/ users/steve/ipportfw/linux21/.

- IP: Masquerading

  Zdokonalená maškaráda v jádru 2.2. Poskytuje další podporu maskovacím protokolům atd. Další informace naleznete v návodu k IP Chains.

Zařízení jádra

V Linuxu existuje několik blokových a znakových zařízení, která také souvisejí s bezpečností.
Jádro obsahuje zařízení /dev/random a */dev/urandom*, která slouží jako generátor náhodných čísel. Obě zařízení by měla být dostatečně bezpečná pro použití při generování PGP klíčů, ssh výměny
a dalších aplikací, kde se vyžadují bezpečně náhodná čísla. Útočník nesmí být schopen ze zna-losti části náhodné sekvence odhadnout její další pokračování. Bylo vynaloženo značné úsilí k zajištění, že čísla generovaná těmito zařízeními jsou opravdu náhodná v pravém slova smyslu.

Rozdíl mezi těmito zařízeními spočívá v tom, že pokud generátoru */dev/random* dojdou náhod-ná čísla, čeká, než nashromáždí další. Na některých systémech to může vést k zablokování na delší dobu, než se nasbírá dostatek uživatelem generované „entropie“. Zařízení /dev/random je tedy nutné používat s rozvahou. (Pravděpodobně nejlepší je použít toto zařízení při generování velmi citlivých klíčů, kdy uživateli řeknete, ať ťuká do klávesnice, hýbe myší, dokud neřeknete Dost!)

Zařízení */dev/random* používá kvalitní zdroj entropie založený na měření intervalů mezi přeruše-
ními a na dalších zdrojích. Blokuje se, dokud není shromážděn dostatečný počet náhodných bitů. Zařízení /dev/urandom je podobné, pokud ale nemá dostatek entropie, vrátí silný kryptografický hash toho, co je k dispozici. Není to tak bezpečné, ale pro většinu aplikací to stačí.

Z těchto zařízení můžete číst například příkazem:

```
root# head -c 6 /dev/urandom | mimencode
```

Tím získáte šest náhodných znaků, které můžete použít jako heslo. Program mimencode nalez-nete v balíčku metamail.Popis algoritmu naleznete v souboru /usr/src/linux/drivers/char/random.c.

Bezpečnost sítě

Význam zabezpečení sítě roste s tím, že uživatelé tráví stále více a více času připojeni. Narušení síťové bezpečnosti bývá často mnohem snadnější než narušení bezpečnosti fyzické nebo lokální a je také mnohem běžnější.

K zajištění síťové bezpečnosti existuje řada dobrých nástrojů, přičemž stále větší množství se stává standardní součástí linuxových distribucí.

Odposlech paketů

Jeden z nejobvyklejších způsobů, jak útočník získá přístup k většímu množství systémů, je insta-lace programu pro odposlech paketů na počítači, do nějž se mu už podařilo proniknout. Tyto takzvané *sniffery* nedělají nic jiného, než že na síťovém rozhraní poslouchají a snaží se v pake-tech najít věci jako passwd, login nebo su a pak zaznamenají následující komunikaci. Tímto způ-sobem útočník získá hesla k systémům, které ani nemusí napadat. Velmi zranitelná jsou tímto způ-sobem hesla posílaná v přímém tvaru.
Příklad: Počítač A byl napaden. Útočník nainstaloval sniffer. Ten zachytil přihlášení administráto-ra na stroj B ze stroje C. Tím útočník získal heslo administrátora ke stroji B. Následně administrá-tor zadal příkaz su, aby mohl provést nějakou správu systému. Tím má útočník heslo superuživa-tele stroje B. Později administrátor nechá někoho ze svého účtu přihlásit se ke stroji Z na úplně jiné síti. Tím má útočník uživatelské jméno a heslo na stroj Z.

V době strukturované kabeláže už ani není nutné, aby útočník nejprve musel nějaký systém
napadnout. Stačí mu přinést si laptop a někde v budově se napojit do sítě.Spolehlivá obrana proti těmto útokům je použití ssh a dalších metod šifrování hesla. Další obra-nou je například protokol APOP pro výběr poštovních schránek. (Klasický protokol POP je velmizranitelný, protože posílá po síti přímo nešifrovaná hesla.)

Systémové služby a tcp_wrapper

Než linuxový systém připojíte k *jakékoliv* síti, nejprve se podívejte, které služby musíte poskyto-vat. Služby, které poskytovat nemusíte, by měly být vždy vypnuté – máte tak o starost méně a útočník má o možnost méně, jak nalézt nějakou díru.

V Linuxu existuje celá řada způsobů, jak vypínat služby. Podívejte se do konfiguračního souboru /etc/inetd.conf a uvidíte, které

síťové služby nabízí démon *inetd*. Všechny nepotřebné zako-mentujte (na začátek příslušného řádku zadejte znak #) a poté pošlete démonu inetd signál SIGHUP.

Dále můžete odstranit (nebo zakomentovat) služby v souboru /etc/services. Povede to k tomu, že danou službu nebudou moci použít ani lokální uživatelé (pokud například odstraníte službu ftp a uživatel se z daného počítače pokusí o FTP připojení na vzdálený počítač, nezdaří se mu to a obdrží chybové hlášení „unknown service“). Obvykle ale nestojí za to služby z /etc/services odstraňovat, protože to bezpečnost nijak nezvyšuje. Pokud by lokální uživatel chtěl použít službu FTP i přesto, že je zakomentována, stačí mu použít vlastního klienta FTP a vše bude fungovat.

Služby, které obvykle budete nechávat zapnuty, jsou:

ftp,
telnet (nebo ssh),
pošta, například pop-3 nebo imap,
identd,

Pokud víte, že nějaký balíček vůbec nebudete používat, můžete jej odstranit celý – v distribucích RedHat to provedete příkazem rpm -e balíček. V distribucích Debian udělá to samé příkaz dpkg -remove.

V každém případě byste měli v souboru /etc/inetd.conf vypnout nástroje rsh/rlogin/rcp, tedy služby login, shell a exec. Tyto protokoly jsou extrémně nebezpečné a v minulosti byly příčinou celé řady úspěšných útoků.

Dále byste měli zkontrolovat adresáře /etc/rc.d/rc[0-9].d (na RedHatu, v Debianu je to /etc/rc[0-9].d) a podívat se, zda se zde nespouštějí nějaké nepotřebné servery. Soubory v tomto adresáři jsou fakticky symbolické odkazy na soubory v /etc/rc.d/init.d (na RedHatu, v Debianu /etc/init.d). Přejmenováním souboru v adresáři init.d zrušíte všechny symbolic-ké odkazy na něj. Pokud chcete nějakou služby vypnout pouze na určité úrovni běhu, přejme-nujte příslušný odkaz nahrazením velkého *S* malým *s*, takto:

root# cd /etc/rc6.d root# mv S45dhcpd s45dhcpd
Používáte-li rc soubory v uspořádání BSD, hledejte nepotřebné programy v /etc/rc*. Většina linuxových distribucí se dodává s tzv. tcp_wrappery, které „obalují“ všechny služby TCP. Tcp_wrapper (*tcpd*) se spouští ze souboru inetd namísto skutečného serveru. Pak provede kon-trolu, zda konkrétní počítač má právo požadovat konkrétní službu, a buď spustí server, anebo při-pojení odmítne. Pomocí programu tcpd tak můžete omezit přístup ke službám TCP. Měli byste
vytvořit soubor /etc/hosts.allow a přidat do něj pouze ty počítače, kterým chcete přístup ke službám povolit. Jste-li klasický domácí uživatel, doporučujeme vám zakázat všechny služby. Navíc tcpd zazname-
nává neúspěšné pokusy o přístup ke službám, takže poznáte, když se někdo snaží o útok. Přidá-váte-li nové služby založené na protokolu TCP, měli byste je nakonfigurovat tak, aby používaly tcp_wrapper. Domácí uživatelé mohou zabránit ostatním v připojení se k jejich počítači, a přitom stále budou mít možnost přijímat poštu a navazovat spojení na Internet. Stačí v souboru /etc/hosts.allow nastavit:

ALL: 127.

A v souboru /etc/hosts.deny samozřejmě nastavit:

ALL: ALL

Tím se zabrání v připojení k vašemu počítači zvenčí, ale stále máte umožněno připojovat se na

Internet.Nezapomeňte, že tcp_wrapper chrání pouze služby spouštěné démonem inetd a několik málo dal-ších. Stále však mohou na vašem počítači běžet i jiné služby. Příkazem netstat -ta můžete zjistit,jaké služby váš počítač nabízí.

Kontrola informací v DNS

Udržování aktuálních informací DNS o všech počítačích ve vaší síti vede ke zvýšení bezpečnosti. Pokud se do vaší sítě připojí neautorizovaný počítač, můžete jej poznat podle toho, že nemá plat-ný záznam DNS. Řadu služeb je možné nakonfigurovat tak, aby nepřijímaly spojení od počítačů, které nemají platné záznamy DNS.

Identd

Identd je malý program typicky spouštěný serverem inetd. Sleduje, který uživatel má spuštěnu
jakou TCP službu, a hlásí to tomu, kdo o tyto údaje požádá.Řada lidí nechápe užitečnost služby identd, a proto ji vypínají anebo blokují všechny dotazy. Služ-ba identd nepomáhá vzdáleným systémům. Neexistuje způsob, jak zjistit, zda údaje poskytnutétouto službou jsou pravdivé. Dotazy také neumožňují žádnou autentizaci.

K čemu tedy taková služba je? Pomáhá vám a představuje další monitorovací nástroj. Pokud služ-ba identd funguje správně, pak víte, že vzdáleným systémům odesílá uživatelské jméno nebo uiduživatelů, kteří používají TCP služby. Obrátí-li se na vás správce vzdáleného systému a řekne vám,že uživatel ten/a/ten chtěl napadnout jejich systém, můžete proti němu snadno zakročit. Pokudslužbu identd nepoužíváte, budete muset projít spousty a spousty logů, zjistit, kdo byl zrovna při-hlášen, a obecně vám to

bude trvat mnohem déle, pokud vůbec budete úspěšní.

Program identd, dodávaný s většinou distribucí, se dá nastavovat mnohem více, než většina lidí tuší. Pro určité uživatele jej můžete vypnout (prostřednictvím souboru .noident), můžete zazna-menávat všechny příchozí dotazy (doporučujeme), můžete dokonce říct, že namísto uživatelské-ho jména má posílat uid nebo text NO-USER.

Konfigurace a zabezpečení MTA Postfix

Postfix je poštovní server napsaný Wietsem Venemou, autorem řady bezpečnostních produktů, jako pokus poskytnout alternativu ke všeobecně rozšířenému poštovnímu programu *sendmail.* Postfix usiluje o to „být rychlý, snadno spravovatelný a snad i bezpečný a zároveň dostatečně kompatibilní se sendmailem, aby uživatelům nevadil".

Další informace o programu Postfix můžete najít na Postfix home a na Configuring and Securing Postfix.

SATAN, ISS a další síové skenery

Existuje celá řada různých softwarových balíků, které slouží ke skenování portů a služeb u počí-tačů v síti. Mezi ty nejznámější patří SATAN, ISS, SAINT a Nessus. Tyto programy se připojují k cílo-vému počítači (nebo k cílovým počítačům) na všech portech, na kterých to jde, a snaží se zjistit, jaké služby zde běží. Na základě těchto informací pak můžete říct, zda je počítač napadnutelný určitým typem útoku.

SATAN (Security Administrator's Tools for Analyzing Networks) je port skener s webovým roz-hraním. Lze jej nastavit na provádění jednoduché, střední nebo silné kontroly počítače nebo počítačů v síti. Je rozumné si tento program opatřit a zkontrolovat jím počítače ve vaší síti a pak odstranit nalezené problémy. Ujistěte se, že vaše kopie programu pochází přímo z metalabs (*http://metalab.unc.edu/pub/packages/security/Satan-for-Linux/*) nebo z důvěryhodného serveru. Na Internetu se objevil i stejnojmenný trojský kůň (*http://www.trouble.org/~zen/satan/satan.html*). Navíc SATAN se už delší dobu nevyvíjí, takže jiné nástroje mohou posloužit lépe.

Dalším port skenerem je ISS (Internet Security Scanner). Je rychlejší než SATAN, a tedy vhodněj
ší pro velké sítě. SATAN nicméně poskytuje více informací.Abacus je sada nástrojů k zajištění bezpečnosti systému a detekci průniků. Další informace zjistí-te na domovské stránce programu na adrese *http://www.psionic.com/abacus.*

SAINT je aktualizovanou verzí skeneru SATAN. Má webové rozhraní a nabízí mnohem víc aktuál
ních testů než SATAN. Další informace viz *http://www.wwdsi.com/~saint.*Dalším skenerem je Nessus. Má grafické rozhraní a umožňuje přidávat doplňky pro prováděnínových typů testů. Další informace najdete na adrese *http://www.nessus.org/.*

Detekce skenování portů

Existují nástroje, které vás upozorní, pokud se někdo pokouší na váš počítač použít SATAN, ISS nebo jiné skenovací nástroje. Pokud ale používáte tcp_wrapper a pravidelně kontrolujete logova-cí soubory, měli byste takové pokusy sami zaznamenat. I při nejšetrnějším nastavení SATAN stále na běžném RedHat systému zanechá v logovacích souborech záznamy o své činnosti.

Existují také „neviditelné" skenery. Paket s nastaveným bitem TCP ACK (který se používá u navá-zaného spojení) pravděpodobně projde firewallem. Navrácený paket RST z portu, *na němž žádné spojení není navázáno*, se pak dá považovat za důkaz „života" na tomto portu. Takovou aktivitu tcp_wrapper nezachytí.

Mohl by vás zajímat program SNORT, který detekuje různé typy síťových útoků, viz *http://www.snort.org/.*

Sendmail, qmail a další MTA

Jednou z nejdůležitějších poskytovaných služeb je poštovní server. Bohužel je tento server záro-veň nejzranitelnější vzhledem k množství úkonů, které musí vykonávat, a k privilegiím, jež k tomu potřebuje.
Pokud používáte *sendmail*, je nesmírně důležité používat aktuální verzi. Tento program má veli-ce bohatou historii různých odhalených chyb. Vždy se ujistěte, že používáte nejnovější verzi, z *http://www.sendmail.org.*

Nezapomeňte, že pokud chcete jenom odesílat poštu, pak sendmail nepotřebujete. Jako běžný domácí uživatel můžete sendmail klidně vypnout a poštu odesílat přímo poštovním klientem. Můžete také odstranit parametry -bd ze spouštěcího souboru programu sendmail, takže nebude přijímat připojení ze sítě. Jinak řečeno, stačí v příslušném spouštěcím skriptu spouštět sendmail pouze takto:

```
# /usr/lib/sendmail -q15m
```

Tím zajistíte, že sendmail bude každých 15 minut kontrolovat lokální frontu zpráv a odesílat zprá-
vy v ní uložené.Řada administrátorů sendmail vůbec nepoužívá a místo toho volí jiné programy pro zpracovánípošty.
Doporučujeme například *qmail.* Ten byl od počátku navrhován s ohledem na bezpečnost.Je rychlý, stabilní a bezpečný. Naleznete jej na adrese *http://www.qmail.org.*

Přímým konkurentem qmailu je *postfix* Wietse Venemy, autora programu tcp_wrapper a dalších bezpečnostních produktů. Původně se jmenoval *vmailer* a jeho vývoj sponzorovala společnost IBM. Byl rovněž od počátku navrhován s ohledem na

bezpečnost. Další informace o tomto pro-gramu najdete na adrese *http://www.postfix.org.*

Útoky typu odepření služeb

Útok typu *odepření služeb* (DoS) je typ útoku, kdy se útočník snaží natolik zatížit nějaký prostře-dek, že nebude schopen reagovat na oprávněné požadavky, nebo kdy se snaží oprávněným uži-vatelům úplně znemožnit přístup k počítači.

DoS útoky se v posledních letech hodně rozšiřují. Dále uvádíme některé nejznámější a nejnověj-ší. Kromě toho se stále objevují nové typy těchto útoků. Aktuální informace naleznete v konfe-rencích věnovaných bezpečnosti a v konferenci bugtraq.

SYN Flooding – jedná se o síťový útok DoS. Využívá mechanismu, jakým je implemento-váno navazování spojení TCP. Nová linuxová jádra (2.0.30 a vyšší) obsahují různé volby, které umožní zabránit tomuto útoku v zablokování přístupu k počítači a službám. Přísluš-né parametry jádra jsou popsány v kapitole „Bezpečnost jádra".
Pentium „F00F" bug – ukázalo se, že jistá sekvence strojových instrukcí způsobí restart kla-sických procesorů Pentium, bez ohledu na provozovaný operační systém. Týká se to pouze klasických Pentií, nikoliv modernějších verzí (Pentium Pro, Pentium II a vyšší). Jádra 2.0.32 a vyšší obsahují speciální ochranu proti tomuto útoku, která navíc byla v jádře 2.0.33 vylep-šena.
Ping Flooding – jde o jednoduchý útok „hrubou silou". Útočník posílá záplavu paketů ICMP. Pokud tento útok pochází z počítače s lepší konektivitou, než máte vy, váš počítač nebude schopen normální komunikace po síti. Varianta tohoto útoků, tzv. *smurfing*, posí-lá pakety ICMP jinému počítači a jako odesilatele uvádí váš počítač, takže původce útoku je hůře detekovatelný. Další informace o tomto typu útoku můžete najít na adrese

http://www.quadrunner.com/~chuegen/smurf.txt.

Pokud takovýto útok zjistíte, pomocí programu tcpdump nebo podobného zjistěte, odkud útok pochází (nebo odkud se *tváří*, že pochází), a sdělte to svému poskytovateli připojení. Tento typ útoku se dá velmi snadno zablokovat na připojovacím routeru nebo firewallu.
Ping of Death – tento útok posílá pakety ICMP ECHO REQUEST příliš dlouhé, než aby se vešly do datových struktur, které jsou pro ně určeny. Protože posláním jediného velkého (65 510 bajtů) paketu je možné řadu systémů zablokovat, nebo dokonce zhroutit, byl útok záhy pojmenován „Ping of Death". Jedná se o známou a dávno opravenou chybu, takže v současné době se jej už nemusíte obávat.
Teardrop/New Tear – poměrně nový útok založený na chybě v implementaci fragmentace protokolu IP v Linuxu i ve Windows. Chyba byla opravena v jádře 2.0.33 a nevyžaduje další volby při překladu. Linux už není tímto druhem útoku zranitelný.

Programy pro jednotlivé útoky a podrobnější popis jejich činnosti můžete najít pomocí vyhledá-vače na adrese *http://www.rootshell.com/.*

Zabezpečení NFS (Network File System)

NFS je velmi rozšířený protokol pro sdílení souborů. Umožňuje, aby servery prostřednictvím démonů *nfsd* a *mountd* „exportovaly" celé souborové systémy na jiné počítače, kde je tento pro-tokol podporován přímo v jádře, nebo nějakým jiným způsobem (nejde-li o linuxové stroje). Démon mountd udržuje informace o připojených souborových systémech v souboru /etc/mtab a zobrazí je příkazem showmount.

V řadě prostředí se NFS používá k tomu, aby uživatelé měli přístupné své domovské adresáře bez ohledu na to, ke kterému počítači se přihlásí. Zajištění bezpečnosti při exportu je poměrně slabé. Můžete démonu nfsd říct, aby vzdáleného uži-vatele root (*uid=0*) mapoval na uživatele nobody, čímž mu zabráníte v úplném přístupu k expor-tovanému souborovému systému. Protože však jednotliví uživatelé mají přístup ke svým soubo-rům (nebo přesněji k souborům s jejich uid), může vzdálený root provést su na účet libovolného uživatele a přistupovat tak k jeho souborům. Pro útočníka to představuje pouze malou nepříjem-nost.

Pokud musíte používat NFS, povolte export pouze na ty počítače, kde je to opravdu nutné. Nikdy neexportujte celý kořenový svazek, exportujte pouze ty adresáře, které musíte. Další informace o NFS naleznete v dokumentu NFS-HOWTO na adrese *http://metalab. unc.edu/mdw/HOWTO/NFS-HOWTO.html.*

NIS (Network Information Service), dříve YP

Network Information Service je metoda distribuce informací skupinám počítačů; master NIS udr-žuje informační tabulky a konvertuje je na takzvané mapy NIS. Tyto mapy pak poskytuje po síti a umožňuje klientským počítačům získat přihlašovací jména, hesla, domovské adresáře a další informace (všechny informace ze standardního souboru /etc/passwd). Díky tomu může uživatel změnit heslo jen jednou a změna se projeví v celé doméně NIS.

NIS není bezpečná služba. Nikdy ani tak nebyla navrhována. Byla navržena jako jednoduchá a uži-tečná. Pokud kdokoliv uhodne název vaší domény NIS, může získat kopii souboru passwd a pomocí dalších programů pak luštit hesla uživatelů. Kromě toho je možné NIS i obelstít a pro-vádět spoustu nehezkých triků. Pokud musíte NIS používat, berte na vědomí všechna s tím spo-jená

rizika.

Existuje i bezpečnější varianta služby NIS, pojmenovaná NIS+. Podrobnější informace najdete v dokumentu NIS-HOWTO na adrese *http://metalab.unc.edu/mdw/HOWTO/NIS-HOWTO.html.*

Firewally

Firewall představuje prostředek k řízení toho, jaké informace mohou cestovat dovnitř a vně vaší sítě. Typicky je firewall připojen jednak k Internetu a jednak k lokální síti a jakákoliv komunika-ce s Internetem je možná pouze přes něj. Díky tomu může firewall určovat, co může mezi vaší sítí a Internetem putovat.

Existuje řada typů firewallů a způsobů, jak je nastavit. Velmi dobrý firewall lze vytvořit z linuxo-vého počítače. Funkce firewallu může být přímo součástí jádra 2.0 a vyšších. Uživatelské nástro-je, jako ipfwadm pro jádra 2.0, ipchains pro jádra 2.2 a iptables pro jádra 2.4, umožňují kdy-koliv nastavovat povolené typy síťového provozu. Různé typy provozu je také možné logovat.

Firewally představují velmi užitečnou a důležitou techniku v zabezpečení sítě. Nikdy však nepod-lehněte dojmu, že jste-li za firewallem, nemusíte se starat o bezpečnost lokálních počítačů. To je fatální chyba. Podrobnější informace o firewallech a Linuxu naleznete ve velmi dobrém doku-mentu Firewall-HOWTO na adrese *http://metalab.unc.edu/mdw/HOWTO/Firewall-HOWTO.html.*

Další informace najdete také v dokumentu IP-Masquerade mini-howto na adrese *http://metalab. unc.edu/mdw/HOWTO/mini/IP-Masquerade.html.*

Podrobnosti o programu ipfwadm (programu, který umožňuje nastavovat firewall) najdete na jeho domovské stránce na adrese *http://www.xos.nl/linux/ipfwadm/.* Pokud nemáte s firewally žádné zkušenosti a hodláte nastavovat více než jen jednoduché zabez-pečení, nutně si přečtěte knihu Firewall vydanou nakladatelstvím O'Reilly and Associates (*http://www.ora.com*) nebo nějakou jinou literaturu věnovanou této problematice. Vynikající doku-menty o firewallech zveřejnil National Institute of Standards and Technology. I když pocházejí z roku 1995, stále jsou velmi aktuální. Najdete je na adrese *http://csrc.nist.gov/nistpubs/800-10/main.html.* Další zajímavé odkazy jsou:

The Freefire Project – seznam zdarma dostupných firewallů, *http://sites.inka.de/sites/lina/ freefire-l/index_en.html.*
SunWorld Firewall Design – dokument stejných autorů jako výše zmíněná kniha naklada-telství O'Reilly. Popisuje různé typy firewallů. Najdete jej na adrese *http://www. sunworld.com/swol-01-1996/swol-01-firewall.html.*
Mason – nástroj pro automatické nastavení firewallu v Linuxu. Tento skript se postupně učí, co potřebujete na síti dělat. Další informace najdete na adrese *http://www.pobox.com/ ~wstearns/mason/.*

IP Chains – firewall v jádře 2.2

Linux IP Firewalling Chains je aktualizace firewallového kódu z jádra 2.0 v jádrech 2.2. Oproti předchozím implementacím obsahuje řadu vylepšení, například:

Pružnější manipulace s pakety.
Lepší možnosti účtování.
Atomické změny nastavení.
Možnost explicitního zpracování fragmentů.
Logování podezřelých paketů.
Obsluha i jiných protokolů než ICMP/TCP/UDP.

Pokud používáte program ipfwadm na jádře 2.0, existují skripty, které provedou konverzi konfi-guračních souborů do formátu ipchains.
Další informace naleznete v dokumentu IP-Chains-HOWTO (*http://www.adelaide.net.au/ ~rustcorp/ipfwchains/ipfwchains.html*).

Netfilter – firewall v jádře 2.4

Jedná se o další vylepšení jaderného firewallu pro jádra 2.4.Subsystém netfilter představuje kompletně přepracované předchozí implementace ipfwadm
a ipchains. Netfilter nabízí celou řadu vylepšení a představuje tak velmi robustní a spolehlivýnástroj pro zabezpečení sítí.Nastavování pravidel firewallu se provádí příkazem iptables.Netfilter umožňuje manipulovat s pakety při jejich průchodu různými částmi jádra. Jednotlivé části
obsahují podporu maškarády, klasický paketový filtr a podporu složitějších funkcí, jako je NAT.
Systém obsahuje dokonce podporu pro distribuci zátěže určitého typu požadavků na více serve
rů.

Velmi mocné jsou funkce stavové inspekce. Stavová inspekce umožňuje sledovat a řídit komuni-kaci procházející filtrem. Díky

možnosti udržovat přehled stavových a kontextových informací

o jednotlivých spojeních se usnadňuje vytváření pravidel pro interpretaci protokolů vyšší úrovně.
Navíc je možné vytvářet samostatné moduly plnící další funkce, například předání paketů uživa

telskému programu a jejich následné vrácení do jádra. Díky tomu se zjednodušují operace, kterébylo dříve nutné řešit přímými zásahy do jádra.Další informace o IP Tables naleznete v dokumentech:

Oskar Andreasson IP Tables Tutorial (*http://www.linuxsecurity.com/feature_stories/feature_ story-94.html*) – rozhovor serveru *LinuxSecurity.com* s Oskarem Andreassonem o jeho „IP Tables tutorialu“ a o použití tohoto dokumentu při vytváření robustních firewallů.
Linux Security Quick-Start (*http://www.linuxsecurity.com/feature_stories/feature_story-93.html*) – Hal Burgiss je autorem dvou základních příruček věnovaných zabezpečení Linu-xu, kde se mimo jiné popisuje i správa firewallu.
Netfiltering Homepage (*http://netfilter.samba.org/*) – domovská stránka produktu.
Linux Kernel 2.4 Firewalling Matures: netfilter (*http://www.linuxsecurity.com/feature_ stories/kernel-netfilter.html*) – tento článek na serveru *LinuxSecurity.com* obsahuje základ-ní informace o filtraci paketů, o tom, jak používat iptables, a nové funkce poslední gene-race linuxových firewallů.
Praktickou adresou pro české uživatele může být *http://www.petricek.cz/mpfw/* s ukázkami konfigurace firewallu.

VPN – Virtuální privátní sítě

VPN je metoda umožňující vytvořit „virtuální“ síť nad nějakou stávající sítí. Tato virtuální síť je typicky šifrovaná a předává data pouze mezi známými entitami připojenými k síti. VPN se často používají k připojení do interní firemní sítě zvnějšku přes Internet.
Pokud používáte maškarádovací firewall v Linuxu a potřebujete povolit průchod paketů MS PPTP (VPN produkt společnosti Microsoft), existuje úprava jádra, která to umožňuje, viz ip-masq-vpn. Pro Linux existuje několik produktů VPN:

vpnd, *http://sunsite.dk/vpnd/*
Free S/Wan, *http://www.xs4all.nl/~freeswan/*

Implementace VPN pomocí ssh, více viz VPN mini-howto
vps (virtual private server), *http://www.strongcrypto.com/*
yawipin, mailto:*http://yavipin.sourceforge.net*

Další odkazy najdete také v části věnované IPSEC.

Bezpečnostní příprava (než se připojíte k síti)

Máte tedy počítač zkontrolován, věříte, že je tak bezpečný, jak je jen možné, a chystáte se jej při-pojit k síti. Existuje několik věcí, které byste měli v tomto okamžiku udělat jako ochranu před úto-kem, abyste byli schopni útok rychle detekovat, zastavit jej a počítač znovu zprovoznit.

Pořite si úplnou zálohu

Debata o metodách a způsobech zálohování je mimo rozsah tohoto dokumentu, uveďme si však

několik informací vztahujících se k zálohování a bezpečnosti.Pokud máte na disku uloženo méně než 650 MB dat, dobrá metoda je vytvořit zálohu na CD disk.Takovou zálohu nelze později modifikovat a při dobrém uložení má dlouhou životnost. Samo-zřejmě budete potřebovat alespoň 650 MB volného diskového prostoru k vytvoření obrazu zálo-hovaných dat. Pásky a další přepisovatelná média byste měli ihned po dokončení zálohování chrá-nit proti zápisu. Nezapomeňte zálohy uložit na bezpečné místo. Dobrá záloha zajistí, že máte bod,od nějž můžete systém obnovovat.

Zvolte vhodný zálohovací plán

Velmi snadno se používá cyklus šesti pásek. Pracuje se čtyřmi páskami pro každý pracovní den, pátou pro sudé pátky a šestou pro liché pátky. Každý den se provádí inkrementální záloha, každý pátek pak úplná záloha na příslušnou pásku. Pokud provedete nějaké zásadní změny konfigurace nebo doplníte důležitá data, je rozumné pořídit úplnou zálohu mimo běžný páskový cyklus.

Testujte zálohy

Pravidelně zálohy testujte, abyste měli jistotu, že fungují tak, jak čekáte. Pravidelně byste měli soubory obnovit a porovnat s ostrými daty, dále byste měli pravidelně testovat čitelnost starších záloh.

Zálohujte databáze RPM

V případě napadení počítače můžete databázi RPM použít podobně jako program *tripwire*, ovšem pouze máte-li jistotu, že nebyla modifikována. Měli byste mít zálohu databáze RPM uloženu na disketě mimo počítač. Podobné funkce lze použít i na distribucích Debian.

Soubory /var/lib/rpm/fileindes.rpm a /var/lib/rpm/packages.rpm se velmi pravděpo-dobně nevejdou na jednu disketu. Pokud je ale zkomprimujete, můžete každý nahrát na samo-statnou disketu.

Při napadení systému pak můžete použít příkaz:

```
root# rpm -Va
```

a provést kontrolu jednotlivých souborů v počítači. Podívejte se na manuálové stránky programu rpm, protože program nabízí ještě další volby. Musíte také mít zajištěno, že nedošlo přímo k modifikaci programu rpm.
Při přidání každého balíčku RPM do systému musíte provést zálohu databáze RPM. Sami se rozhodněte, zda to za to stojí.

Sledujte logy systému

Důležité je, aby nedošlo k porušení systémových logů. Dobrý začátek je nastavit soubory v adre
sáři /var/log tak, aby je mohla číst a zapisovat pouze omezená skupina uživatelů. Nezapomínejte pravidelně sledovat, co se v těchto záznamech objevuje. Důležité jsou typicky zá-znamy třídy „auth“ – například vícenásobné neúspěšné přihlášení může indikovat pokus o útok.

Umístění logovacích souborů závisí na distribuci. V systémech, které odpovídají standardu „Linux Filesystem Standard“, jako je například RedHat, hledejte v adresáři /var/log soubory messages, maillog a další.

Kam ukládá logovací soubory ta která distribuce, můžete zjistit v souboru /etc/syslog.conf.
Tento soubor říká logovacímu démonu syslogd, kam záznamy ukládat.
Kromě toho můžete nakonfigurovat skript nebo démona zajišťující rotaci logů tak, abyste měli více
času na jejich prozkoumání. V nových distribucích RedHat to zajišťuje balíček logrotate, jiné distri
buce mají jistě něco podobného.

Jestliže došlo k poškození logů, zkuste, zda se vám podaří zjistit, kdy k poškození došlo a co bylo změněno. Existují delší časová období, pro která nejsou žádné záznamy? Dobrý nápad je vyhle-dat nepoškozené logy v zálohách (pokud je máte).

Typicky útočník modifikuje logovací soubory, aby zakryl stopy po své činnosti, a právě proto byste neměli zapomínat je pravidelně kontrolovat. Můžete zaznamenat pokus útočníka o přístup nebo objevit program, který se snaží o přístup na účet superuživatele. Možná se vám podaří odhalit podezřelé záznamy dříve, než je útočník zlikviduje.

Rozhodně byste měli záznamy třídy *auth* oddělit od ostatních záznamů – týká se to především zá-znamů o použití příkazu su, záznamů o přihlášení uživatelů a dalších informací o uživatelských účtech.

Pokud je to možné, nakonfigurujte syslogd tak, aby kopii důležitých dat posílal na bezpečný systém. Tím se útočníkovi zabrání zakrýt důkazy o své přítomnosti. Podrobnosti viz manuálová stránka syslogd.conf, volba @.

Existují i propracovanější logovací programy. Podívejte se na program Secure Syslog na adrese *http://www.core-sdi.com/ssyslog/*. Tento program umožňuje šifrovat záznamy v logu a zabránit tak jejich modifikaci.

Dalším takovým programem je syslog-ng (*http://www.balabit.hu/products/syslog-ng.html*). Nabízí podstatně větší možnosti logování a podporuje záznam na vzdálený systém, aby byly logy chrá-něny před modifikací.

A konečně – logovací záznamy jsou k ničemu, pokud je nikdo nečte. Udělejte si občas chvíli času a podívejte se na ně, abyste získali představu, jak vypadají za normálních okolností. Daleko snáze pak odhalíte cokoliv neobvyklého.

Provádějte aktualizace systému

Většina uživatelů instaluje Linux z CD disků. Vzhledem k průběžnému odhalování bezpečnostních problémů dochází k častému zveřejňování nových (opravených) programů. Než počítač připojíte k síti, je rozumné navštívit stránky distributora a stáhnout si aktualizované verze programů, které jste nainstalovali z CD. Často tyto nové verze obsahují důležité bezpečnostní opravy, a je tedy velmi rozumné je nainstalovat.

Co dělat během a po útoku

Řekněme, že i když jste se drželi zde (nebo jinde) popsaných doporučení, zaznamenali jste útok na váš systém. Co teď?
Nejdůležitější je zůstat klidný. Neuváženým postupem můžete napáchat více škody než samotný útočník.

Útok probíhá

Odhalení právě probíhajícího útoku může být docela vzrušující. Vaše reakce může mít dalekosáhlé

následky.Pokud jde o fyzické narušení, zjevně jste přistihli někoho; může jít přímo o fyzické vloupání k vámdomů, do kanceláře nebo do učebny. Pak je vhodné obrátit se na příslušné instituce. V případěučebny může jít také o to, že se někdo snaží otevřít nebo restartovat počítač. V závislosti na vašípozici a zavedených postupech jej můžete požádat o ukončení jeho činnosti nebo se můžete obrá-tit na ostrahu.

Odhalíte-li útok ze strany lokálního uživatele, nejprve se ujistěte, že útočí opravdu ten, koho mys-líte. Podívejte se, odkud je uživatel přihlášen. Přihlašuje se takto běžně? Ne? Zkuste jej kontakto-vat nějakým nepočítačovým způsobem – například telefonicky nebo zajděte do jeho kanceláře.Pokud je opravdu přihlášen, požádejte jej o vysvětlení, co že to dělá, nebo jej rovnou požádejte,ať s danou činností přestane. Pokud není přihlášen a nemá potuchy, o čem mluvíte, pak je nutnésituaci podrobněji prověřit. Než vznesete nějaké obvinění, shromážděte dostatečné množství spo-lehlivých informací.

Detekujete-li síťový útok, pak v první řadě (je-li to možné) odpojte počítač od sítě. Je-li připojenmodemem, odpojte telefonní kabel, je-li připojen přes Ethernet, odpojte ethernetový kabel. Tímzabráníte vzniku dalších škod a útočník se bude domnívat, že jde nejspíš o síťový problém, a neo odhalení útoku.

Nemůžete-li počítač odpojit od sítě (jedná se o důležitý systém nebo k němu nemáte fyzický pří

stup), použijte nástroje jako tcp_wrapper nebo ipfwadm a zablokujte přístup útočícího systému. Pokud nemůžete zablokovat přístup celému vzdálenému systému, zablokujte uživatelský účet,přes nějž útok probíhá. Nezapomeňte, že zablokování účtu není úplně triviální. Nezapomínejte nasoubory .rhosts, přístup přes FTP a případná možná zadní vrátka.

Jakmile provedete výše popsané (odpojení systému, zablokování přístupu, zablokování účtu),

zabijte všechny procesy daného uživatele a odhlaste jej.Několik dalších minut byste měli systém pozorně sledovat, protože útočník se bude pravděpo-dobně chtít vrátit – možná použije jiný účet, možná se připojí z jiné adresy.

K útoku už došlo

Odhalili jste útok nebo se vám jej podařilo přímo zastavit? Takže co teď?

Zavření přístupu

Pokud se vám podaří odhalit metodu, kterou se útočník do počítače dostal, měli byste tuto přístupovou cestu odříznout. Můžete si například všimnout neobvyklé aktivity přes FTP bezprostředně před přihlášením útočníka. Vypněte tedy službu FTP a zkontrolujte, zda neexistuje aktualizovaná verze vámi používaného programu nebo zda nebyla hlášena nějaká chyba.

Zkontrolujte logy a podívejte se na stránky s bezpečnostní problematikou, zda nejsou k dispozici nějaké nové opravy. Bezpečnostní opravy pro systém Caldera najdete na stránce *http://www.*

caldera.com/techref/security/. RedHat zatím nemá odděleny bezpečnostní opravy od ostatních
oprav, všechny najdete na adrese *http://www.redhat.com/errata.*Debian má jednak poštovní konferenci věnovanou bezpečnosti, jednak samostatné stránky naadrese *http://www.debian.org/security/.*
Je velmi pravděpodobné, že jakmile se objeví nějaká oprava v jedné distribuci, záhy se objeví
i v dalších.Existuje také projekt bezpečnostního auditu Linuxu. Metodicky prochází jednotlivé uživatelsképrogramy a hledá možné bezpečnostní díry. Z prohlášení projektu:

„Snažíme se o systematický audit zdrojových kódů Linuxu s cílem dosáhnout stejné bezpečnosti jako u OpenBSD. Doposud jsme odhalili (a opravili) několik problémů, uvítáme však další pomoc. Konference není moderovaná a může sloužit jako dobrý zdroj obecných informací o bezpečnos-ti. Adresa je *security-audit@ferret.lmh.ox.ac.uk.* Chcete-li se do konference přihlásit, pošlete zprá-vu na adresu *security-audit-subscribe@ferret.lmh.ox.ac.uk.*“

Pokud se vám nepodaří útočníkovi zablokovat přístup, velmi pravděpodobně se vrátí – a to nejen přímo do napadnutého počítače, ale možná i jinam do sítě. Pokud se útočníkovi podařilo nějakou dobu odposlouchávat provoz na síti, je velmi pravděpodobné, že získal přístup i k jiným systémům.

Odhad škody

Prvním krokem je odhad škody. Co bylo poškozeno? Pokud používáte nějakou metodu kontroly integrity, například Tripwire, můžete zkontrolovat, které soubory byly poškozeny. Pokud nic takového nemáte, budete muset zkontrolovat všechna důležitá data.

Protože instalace linuxových systémů je poměrně jednoduchá, rozumným řešením může být zálo-ha konfiguračních souborů, smazání disků, reinstalace systému a obnova uživatelských dat a kon-figuračních souborů ze záloh. Tím máte zajištěno, že máte nový, čistý systém. Pokud musíte obno-vovat nějaká data z nabouraného systému, dávejte si obzvláštní pozor na binární soubory, proto-že může jít o trojské koně zanechané útočníkem.

Reinstalace by měla být chápána jako povinná, pokud útočník získal práva superuživatele. Navíc možná budete chtít uchovat důkazy o napadení, takže možná celý disk napadeného systému uložíte na bezpečné místo.

Pak je načase se začít starat o to, jak dlouho napadení trvalo a zda nebyla poškozena i data na zálohách. O problematice zálohování budeme ještě hovořit.

Zálohovat, zálohovat, zálohovat!

Pravidelné zálohy jsou z bezpečnostního pohledu nezbytné. Pokud dojde k nabourání systému, můžete data obnovit ze záloh. Samozřejmě některá data mohou být hodnotná i pro útočníka, takže nejen že vaše data smaže, ale také je ukradne a pořídí si vlastní kopii – tak či tak vám alespoň zbudou zálohy.

Před obnovením poškozených souborů byste měli projít několik záloh zpátky. Útočník mohl sou-bory poškodit už před delší dobou a vy jste mohli úspěšně zálohovat poškozená data! Samozřej-mě i pro zálohy musí platit nějaká bezpečnostní pravidla. Rozhodně je ukládejte na bezpečném místě tak, abyste věděli, kdo k nim má přístup. (Pokud má útočník fyzický přístup k zálohám, může si pořídit kopii dat, aniž byste se to kdy dozvěděli.)

Stopování útočníka

Takže jste útok zastavili a systém obnovili. Tím ale práce nekončí. I když pravděpodobnost vypátrání útočníka není velká, měli byste útok nahlásit.
Útok byste měli ohlásit administrativnímu kontaktu systému, z nějž k útoku došlo. Kontakt může-te zjistit pomocí databáze *whois* nebo Internic. Můžete jim poslat zprávu s relevantními logovací-mi záznamy, daty a časy. Pokud zjistíte o útoku cokoliv dalšího, napište to také. Po odeslání zprá-vy můžete pokračovat telefonátem správci vzdáleného systému. Pokud správce útočníka vypátral, bude s ním možná hovořit a tak dále.

Dobrý útočník často používá mnoho mezilehlých systémů, přičemž některé z nich ani netuší, že jsou takto zneužívány. Pokus o vysledování útočníka může být velmi obtížný. Při rozhovorech se správci jiných systémů buďte zdvořilí, daleko ochotněji vám pak pomohou. Dále byste měli infor-movat případnou bezpečnostní organizaci, jíž jste členem (CERT nebo podobně), a dodavatele distribuce.

Další informace

Informace o bezpečnosti

Existuje velké množství serverů věnovaných obecně bezpečnostním otázkám a speciálně bezpeč-nosti Linuxu. Je rozumné se přihlásit do jedné nebo více bezpečnostních konferencí a sledovat zprávy o nově odhalených chybách. Většina těchto konferencí nemá velký provoz a jejich zprávy jsou velmi cenné.

LinuxSecurity.com

Server LinuxSecurity.com obsahuje velké množství materiálů napsaných pracovníky tohoto serveru a řadou dalších autorů.

Linux Advisory Watch (*http://www.linuxsecurity.com/vuln-newsletter.html*) – vyčerpávající týdeník zaměřený na nově odhalené bezpečnostní chyby. Obsahuje odkazy na aktualizo-vané balíky a popisy jednotlivých chyb.
Linux Security Week (*http://www.linuxsecurity.com/newsletter.html*) – cílem tohoto týdení-ku je poskytnout čtenářům nejdůležitější zprávy z oblasti bezpečnosti Linuxu.
Linux Security Discussion List (*http://www.linuxsecurity.com/general/mailinglists.html*) – poštovní konference zaměřená na obecné otázky bezpečnosti.
Linux Security Newsletters (*http://www.linuxsecurity.com/general/mailinglists.html*) – informace o objednání jednotlivých zpravodajů.
comp.os.linux.security FAQ (*http://www.linuxsecurity.com/docs/colsfaq.html*) – časté otáz-ky a odpovědi z diskusní skupiny comp.os.linux.security.
Linux Security Documentation (*http://www.linuxsecurity.com/docs/*) – vynikající místo s informacemi o bezpečnosti Linuxu a Open-Source produktů.

FTP servery

CERT je Computer Emergency Response Team. Často zveřejňují upozornění na nové útoky a opra
vy. Další informace viz ftp://ftp.cert.org/.ZEDZ (dříve Replay, *http://www.zedz.net/*) nabízí archiv řady bezpečnostních produktů. Protoženejde o americký server, nemusí se řídit americkými omezeními silné kryptografie.

Matt Blaze je autor CFS a odborník na bezpečnost. Jeho archiv najdete na adrese ftp://ftp.rese-arch. att.com/pub/mab. Vynikající nizozemský server věnovaný bezpečnosti je *tue.nl*, ftp://ftp.win.tue.nl/pub/security/.

WWW servery

Hacker FAQ – otázky a odpovědi o hackerech, *http://www.solon.com/~seebs/faqs/ hacker.html.*

Archive COAST obsahuje řadu bezpečnostních programů a informací, *http://www.cs.pur-due.edu/coast/*.
Stránky společnosti SuSE věnované bezpečnosti, *http://www.suse.de/security/*.
Server Rootshell.com ukazuje, jaké díry útočníci momentálně používají, *http://www. rootshell.com/*.
BUGTRAQ zveřejňuje bezpečnostní problémy, *http://www.netspace.org/lsv-archive/ bugtraq.html.*
CERT, Computer Emergency Response Team, oznamuje běžné útoky na unixové systémy,

http://www.cert.org/.

- Dan Farmer je autor programu SATAN a dalších bezpečnostních nástrojů. Jeho stránky obsahují zajímavý průzkum o bezpečnosti a řadu bezpečnostních nástrojů,
  http://www.trouble.org/.
- The Linux Security WWW je server s dobrými informacemi o bezpečnosti Linuxu,
  http://www.aoy.com/Linux/Security/.
- Infilsec je databáze chyb, z níž se dozvíte, jak jsou které systémy zranitelné,
  http://www.infilsec.com/vulnerabilities/.

CIAC vydává pravidelné bulletiny o běžných útocích, *http://ciac.llnl.gov/cgi-bin/index/ bulletins.*
Úvod do technologie PAM najdete na adrese *http://www.kernel.org/pub/linux/libs/pam/*.
Debian nabízí stránky s bezpečnostními opravami a informacemi, *http://www.debian.com/ security/*.
WWW security FAQ Lincolna Steina je vynikající dokument o bezpečnosti webových ser-verů,
http://www.w3.org/Security/Faq/www-security-faq.html.

Poštovní konference

Bugtraq – pro přihlášení se pošlete e-mail na adresu *listserv@netspace.org*, v těle zprávy uveďte text *subscribe bugtraq.* (Archiv konference naleznete na adrese uvedené v před-chozí kapitole.)
CIAC – pošlete e-mail na adresu *majordomo@tholia.llnl.gov*, v těle zprávy uveďte *subscri-be ciac-bulletin*.
RedHat provozuje řadu konferencí, nejdůležitější je *redhat-announce*, kde se můžete dočíst

o bezpečnostních (a jiných) opravách bezprostředně po jejich zveřejnění. Pošlete e-mail na adresu r*edhat-announce-list-request@redhat.com*, jako subject uveďte *Subscribe*. Další informace a archiv najdete na adrese *https://listman.redhat.com/mailman/listinfo/*.

- Debian nabízí konferenci věnovanou bezpečnostním opravám, další informace najdete na adrese *http://www.debian.com/security/*.

Tištěné materiály

Byla vydána celá řada knih věnovaných bezpečnostní problematice. Následující seznam uvádí jen vybrané z nich. Kromě specializovaných knih je otázka bezpečnosti zmiňována i v řadě knih
o správě systému.
D. Brent Chapman, Elizabeth D. Zwicky: Building Internet Firewalls, září 1995, ISBN 156592-124-0
Simson Garfinkel, Gene Spafford: Practical UNIX & Internet Security, duben 1996, ISBN 156592-148-8
Deborah Russel, G. T. Gangemi sr.: Computer Security Basics, červenec 1991, ISBN 0937175-71-4
Olaf Kirch: Linux Network Administrator's Guide, leden 1995, ISBN 1-56592-087-2
Simson Garfinkel: PGP: Pretty Good Privacy, prosinec 1994, ISBN 1-56592-098-8
David Icove, Karl Seger, William von Storch: Computer Crime A Crimefighter's Handbook, srpen 1995, ISBN 1-56592-086-4
John S. Flowers: Linux Security, březen 1999, ISBN 0735700354
Maximum Linux Security: A Hacker's Guide to Protecting Your Linux Server and Network, červenec 1999, ISBN 0471290009
Donn Parker: Fighting Computer Crime, září 1998, ISBN 0471163783

Tištěné materiály v češtině

Bob Toxen: Bezpečnost v Linuxu, Computer Press, Brno, 2003, ISBN 80-7226-716-7
Ramón J. Hontanon: Linux – praktická bezpečnost, Grada, Praha, 2003. ISBN 80-247-0652-0
Andrew Lockhart: Bezpečnost sítí na maximum, Computer Press, Brno, 2005, ISBN 80-2510805-8

Slovníček

Dále uvádíme některé časté pojmy z oboru počítačové bezpečnosti. Vyčerpávající slovník termí-nů z této oblasti naleznete na serveru LinuxSecurity.com.

Authentication – Proces zajištění, že přijatá data jsou stejná jako odeslaná a že odesilatel dat je opravdu tím, za koho se vydává.
Bastion host – Počítačový systém, který musí být silně zabezpečen, protože je vystaven útokům, obvykle proto, že je vystaven na Internetu a představuje hlavní kontaktní systém pro uživatele interní sítě. Jméno dostal po vnějších opevněních středověkých hradů. Z bastionu je možné pozorovat kritické oblasti, obvykle má silné zdi, prostory pro obrán-ce a někdy také koryta, kterými se

lije vařící olej na útočníky.
Buffer overflow – Běžná programátorská chyba je nerezervovat si dostatečný prostor pro data a nekontrolovat jejich velikost. Když dojde k přetečení dat, může být spuštěný pro-gram donucen dělat něco jiného. Typicky se přetečení použije k přepsání návratové adre-sy funkce na zásobníku na jinou adresu.
Denial of Service – Útok, který spotřebuje prostředky systému na něco, na co nejsou urče-ny, takže znemožní použití systému k legitimním účelům.
Dual-homed host – Obecný počítačový systém, který má alespoň dvě síťová rozhraní.
Firewall – Komponenta nebo skupina komponent, které omezují přístup mezi chráněnou síti a Internetem nebo mezi různými sítěmi.
Host – Počítač připojený k síti.
IP spoofing – Složitá technika útoků, sestávající z několika složek. Jedná se o útok, kdy se útočící systém vydává za někoho jiného. Podrobně je tento mechanismus popsán „daemo-nem9“, „routem“ a „infinitym“ ve 48. vydání magazínu Phrack.
Non-repudiation – Schopnost příjemce prokázat, že odesilatel nějaká data odeslal, i když ten by to popíral.
Packet – Základní komunikační jednotka na Internetu.
Packet filtering – Operace, při níž zařízení selektivně řídí tok dat mezi sítěmi. Filtr umož-ňuje propouštět nebo blokovat pakety typicky při jejich směrování z jedné sítě do druhé. Filtrování se nastavuje skupinou pravidel, která říkají, jaké pakety (odkud a kam, na jakých portech) mají být propuštěny a které mají být zahozeny.
Perimeter network – Síť mezi chráněnou sítí a externí sítí, představující dodatečný bezpeč-nostní prvek. Někdy se označuje jako demilitarizovaná zóna, DMZ.
Proxy-server – Program, který komunikuje s externími servery jménem svých klientů. Kli-enti se baví s proxy-serverem, ten jejich požadavky tlumočí externím serverům a vrací kli-entům odpovědi.
Superuser – Označení uživatele root.

Časté otázky

- *Je bezpečnější překládat ovladače přímo do jádra, nebo je překládat jako moduly?*

  Někdo tvrdí, že je lepší zakázat nahrávání ovladačů zařízení jako modulů, protože útočník by mohl namísto požadovaného ovladače nahrát trojského koně nebo modul jinak ovlivňující bezpečnost systému.

  Abyste ale mohli nahrát modul, musíte být superuživatel. Soubory s modulárními ovladači může rovněž přepsat pouze superuživatel. Útočník by tedy k takovému útoku nejprve musel získat práva superuživatele – a pokud se mu to podaří, může udělat podstatně horší věci, než nahrát nějaký modul.

  Moduly slouží k dynamické podpoře zařízení, která nemusí být používána často. Na ser-verech nebo firewallech k takovým situacím obvykle nedochází. Proto je logičtější na ser-veru přeložit potřebné ovladače přímo jako součást jádra. Modulární ovladače jsou navíc pomalejší.

- *Proč se nejde ze vzdáleného systému přihlásit jako root?*

  Viz kapitola „Bezpečnost superuživatele“. Toto omezení je záměrné, aby nebylo možné se aplikací jako telnet přihlásit jako superuživatel. Jednalo by se o závažný bezpečnostní pro-blém, protože heslo se po síti přenáší nešifrovaně. Nezapomínejte: Útočník má na své stra-ně čas a pomocí automatizovaných programů může číhat na zadávaná hesla.

- *Jak zapnu podporu stínových hesel?*

  Jako root spusťte pwconv, tím by měl vzniknout soubor /etc/shadow, se kterým budou aplikace pracovat. Pokud používáte RedHat 4.2 a vyšší, moduly PAM zajistí automatický přechod od souboru /etc/passwd ke stínovým heslům bez nutnosti dalších úprav. Tro-cha doplňujících informací: Stínová hesla jsou mechanismus uložení hesel v jiném soubo-ru než v běžném /etc/passwd. Má to několik výhod. První je ta, že stínový soubor hesel, /etc/shadow, může číst pouze root, zatímco soubor /etc/passwd musí být čitelný všemi uživateli. Další výhoda je, že administrátor může zapínat a vypínat účty, aniž by status kon-krétních účtů byl znám ostatním uživatelům systému.
  Soubor /etc/passwd v tomto případě slouží k uložení uživatelských jmen a skupin a pou-žívají jej programy jako /bin/ls k mapování uživatelského ID na uživatelské jméno ve výpi-sech souborů. Soubor /etc/shadow pak obsahuje pouze jméno uživatele a jeho heslo, a případně některé další informace, například o aktivitě účtu.

  Pokud vás zajímá problematika zabezpečení hesel, bude vás nejspíš zajímat i problemati-ka volby bezpečných hesel. Můžete k tomu použít modul pam_cracklib, který je součástí PAM. Zadaná hesla porovná proti databázi programu Crack a zjistí, jestli je heslo snadno uhodnutelné.

- *Jak můžu zapnout rozšíření SSL pro server Apache?*

a. Opatřete si SSLeay 0.8.0 nebo pozdější.
b. Vytvořte jej, otestujte a nainstalujte!
c. Opatřete si zdrojový kód programu Apache.
d. Z této adresy si stáhněte rozšíření SSLeay programu Apache.
f. Zkonfigurujte je a doplňte do systému.

Můžete také vyzkoušet server ZEDZ.net, na kterém je k dispozici řada připravených balí-ků a nachází se mimo USA.

- *Jak spravovat více uživatelských účtů a stále zachovat bezpečnost?*

  Většina distribucí obsahuje kvalitní nástroje pro nastavování vlastností uživatelských účtů.

Programy pwconv a unpwconv slouží k přechodu na stínová a na normální hesla.
Programy pwck a grpck kontrolují správnost organizace souborů passwd a group.
Programy useradd, usermod a userdel slouží k přidávání, modifikaci a rušení uživatel-ských účtů. Pro práci se skupinami slouží programy groupadd, groupmod a groupdel.
Hesla skupin se nastavují programem gpasswd. Všechny tyto programy umějí praco-vat jak se stínovými, tak i s normálními hesly. Další informace najdete na manuálových stránkách příslušných programů.

- *Jak můžu při použití serveru Apache chránit přístup k některým dokumentům heslem?* Určitě neznáte stránky *http://www.apacheweek.org*, že? Informace o autentizaci uživatelů můžete najít na adrese *http://www.apacheweek.com/-features/userauth*, další informace o bezpečnosti serveru na adrese *http://www.apache. org/docs/misc/security_tips.html.*

Závěr

Budete-li sledovat bezpečnostní konference a pravidelně aktualizovat svůj systém, uděláte pro jeho bezpečnost hodně. Pokud navíc budete sledovat logovací soubory a pravidelně spouštět nástroj typu Tripwire, uděláte ještě více.

Zajištění rozumné úrovně bezpečnosti u domácího počítače není příliš obtížné. Pracnější je to u produkčních strojů, ale i zde Linux poslouží jako bezpečná platforma. Vzhledem k povaze vývo-je Linuxu se bezpečnostní opravy typicky objevují podstatně dříve než u komerčních systémů, což z Linuxu činí optimální platformu pro aplikace, kde se klade důraz na bezpečnost.

Cryptoloop

O tomto dokumentu

Tento dokument HOWTO popisuje, jak lze použít šifrování zařízení pomocí Cryptoloop v řadě jader 2.6 systému Linux. Cryptoloop umožňuje vytvořit šifrované systémy souborů v rámci oddílu nebo jiného souboru v systému souborů. Tyto šifrované soubory lze přesunout na disk CD, DVD, paměťové zařízení USB atd. Cryptoloop používá tzv. loopback zařízení. Jedná se o pseudozařízení fungující jako „smyčka“, přes kterou musí projít každé volání systému souborů. Díky tomu lze zpracovat data, která mají být šifrována a dešifrována. Od verze jádra 2.6 bylo rozhraní Crypto API integrováno do hlavního jádra a nastavení šifrovaného systému souborů se značně usnadnilo. Nejsou požadovány žádné další záplaty jádra. Je však nutné aktualizovat některé uživatelské nástroje. K práci s nástrojem Cryptoloop zatím bohužel neexistuje kvalitní dokumentace. Tento dokument HOWTO byl vytvořen proto, aby všem zájemcům usnadnil vytvoření šifrovaného systému souborů pomocí standardních funkcí nástroje Cryptoloop. Nástroj Cryptoloop je založen na rozhraní Crypto API v jádře 2.6 systému Linux. Neměli byste jej zaměňovat se zařízením Loop-AES, což je zcela nezávislý projekt. Cryptoloop se podobá rozhraní Crypto API, které bylo k dispozici jako samostatná záplata pro jádra řady 2.4. Nová verze není se starou verzí kompatibilní.

Zřeknutí se odpovědnosti

Autoři nepřijímají žádnou odpovědnost za obsah tohoto dokumentu. Použití koncepcí, příkladů a informací je na vaše vlastní riziko. Dokument může obsahovat chyby a nepřesnosti, které mohou způsobit poškození vašeho systému. Postupujte opatrně. Ačkoli je výskyt problémů velmi nepravděpodobný, autoři za ně nemohou přijmout žádnou odpovědnost.

Reakce

Budu velmi rád, když mi pošlete své názory na tento dokument. Své dodatky, komentáře a kriti-ku směrujte na následující e-mailovou adresu: <*cryptoloop@ralfhoelzer.com*>.

Úvod

V současnosti existuje několik alternativ k používání nástroje Cryptoloop. Nejznámější je pravdě-podobně Loop-AES, viz *http://loop-aes.sourceforge.net* nebo další howto v této knize, zaměřené na zašifrování kořenového oddílu. Loop-AES poskytuje velmi podobné funkce jako Cryptoloop. V současnosti je Aes-loop vyspělejší než Cryptoloop a je také rychlejší (podle autora Loop-AES asi dvakrát), protože používá vysoce optimalizovanou implementaci algoritmu AES v jazyce Assem-bler. To neznamená, že nástroj Cryptoloop je pomalý. Při každodenní práci s normálním počtem vstupně-výstupních operací jsem nezaregistroval významný rozdíl v rychlosti mezi oddílem šifro-vaným pomocí nástroje Cryptoloop a nešifrovaným oddílem. Pokud pro vás není výkon vstupně--výstupních operací mimořádně důležitý, měl by vám Cryptoloop vyhovovat. Loop-AES nabízí některé dodatečné funkce, které zatím nejsou dostupné v implementaci nástroje Cryptoloop v jádře. Loop-AES vyžaduje upravené uživatelské nástroje (mount, losetup) a tyto úpravy nejsou s nástrojem Cryptoloop kompatibilní. Nástroje Cryptoloop a Loop-AES nelze používat současně.

Nástroj Cryptoloop je poměrně bezpečný. Klíč je obvykle generován z hesla, jehož hodnota hash slouží jako klíč algoritmu AES. Z toho vyplývá možnost útoku se známým otevřeným textem. Nástroj Loop-AES je z tohoto hlediska dokonalejší, protože generuje náhodný klíč a šifruje jej samostatně. Útok se známým otevřeným textem je díky tomu obtížnější. Loop-AES také podporu-je režim více klíčů, kdy jsou sektory šifrovány pomocí 64 odlišných klíčů AES. Obecně platí, že útok hrubou silou na heslo může být velmi účinný, pokud zvolíte slabé heslo. Chcete-li se proti tomu pojistit, měli byste použít heslo s délkou alespoň 20 znaků. Když to neuděláte, bude útok hrubou silou na heslo mnohem snazší než pokus hrubou silou přímo prolomit šifrování AES.

Funkce Cryptoloop dostupná ve standardním jádře nabízí stabilní a čistou implementaci bez nutnos-ti instalace dodatečných záplat. Protože je tato funkce stále poměrně nová, pravděpodobně nebyla z hlediska bezpečnosti dostatečně zkontrolována. Sami se musíte rozhodnout, co je pro vás vhodné.

Důležité upozornění: Nástroj Cryptoloop byl v nejnovějším jádře 2.6 označen jako nedoporučený (deprecated). To znamená, že nadále nebude aktivně udržován. Nástupcem nástroje Cryptoloop bude dm-crypt (*http://www.saout.de/misc/dm-crypt/*). Dm-crypt je k dispozici v hlavním jádře od verze 2.6.4. Cryptoloop zůstane v hlavním jádře ještě dlouho k dispozici, ale hlavní metodou šif-rování disků bude dm-crypt. Nástroj dm-crypt používá jiný přístup k zařízením (vyžaduje device-mapper) zařízení a poskytuje prakticky stejné funkce jako Cryptoloop. Stále se jedná o naprostou novinku a zatím neexistují žádné snadno použitelné uživatelské nástroje. Kód nástroje dm-crypt by měl být mnohem čistší než kód nástroje Cryptoloop, ale jsou zde některé důležité

rozdíly. Chcete-li například vytvořit šifrovaný systém souborů uvnitř souboru, musíte nadále použít loop--back zařízení, ale podpora této funkce se stále vyvíjí.

K dispozici jsou i další nástroje, které umožňují vytvořit šifrovaný systém souborů. BestCrypt je komerč-ní produkt společnosti Jetico. Umožňuje vytvářet šifrované kontejnery a nabízí velký výběr šifer. Po-skytuje také některé důmyslné funkce, jako jsou skryté kontejnery. Je k dispozici pro systémy Win-dows a Linux, takže se hodí při výměně šifrovaných kontejnerů mezi těmito systémy. BestCrypt lze nyní přeložit i pro jádra řady 2.6. Cryptoloop také umožňuje vytvořit kontejnery, které lze přemisťovat. Stačí vytvořit šifrovaný systém souborů uvnitř souboru, jak je popsáno dále. K souborům, které jsou zašifrovány nástrojem Cryptoloop, podle mých informací nelze přistupovat z jiných operačních systé-mů (např. Windows). V tomto případě můžete být odkázáni pouze na nástroj BestCrypt.

Další program jménem TrueCrypt (*http://www.truecrypt.org/*) umožňuje transparentně šifrovat celé oddíly nebo vytvořit virtuální šifrované disky uvnitř jiných souborů. Podporuje také tvorbu skry-tých svazků a nabízí výběr z mnoha prověřených šifrovacích algoritmů. K dispozici jsou jazykové balíčky včetně češtiny. Od verze 4.0 je program dostupný nejen pro systémy Windows, ale i GNU/Linux. Jedná se o svobodný program typu open-source.

Existují i komerční nástroje na šifrování disků, jako například PGP disk, ale pokud vím, systém Linux je nepodporuje.

Konfigurace jádra

Chcete-li použít nástroj Cryptoloop, musíte aktivovat několik funkcí jádra. Můžete tyto požadova-né funkce buď přeložit jako moduly nebo je přeložit přímo do jádra. Používáte-li základní distri-buční jádro, pravděpodobně budete mít moduly již připravené. Vyzkoušíte to například pomocí modinfo cryptoloop – objeví-li se popis modulu, pak je přeložen a připraven k použití.
Následujícím postupem je zpřístupníte jako moduly. Pokud nemáte zkušenosti s vytvářením jádra řady 2.6, měli byste si přečíst dokument HOWTO o jádře systému Linux (*http://www. linuxdocs.org/HOWTOs/Kernel-HOWTO.html*). Následující pokyny obsahují pouze nejnutnější kroky.

1. Přejděte do adresáře, který obsahuje strom zdrojových souborů jádra (obvykle /usr/src/linux/ – musíte mít nainstalován balíček, např. kernel-source), a spusťte kon-figuraci:

   ```
   make menuconfig
   ```

2. Zapněte podporu obecného loopback zařízení. Aktivujte možnost „Loopback device support“ pod položkou:

   ```
   Device Drivers -> Block Devices -> Loopback device support
   ```

1Ve stejné sekci zapněte podporu nástroje Cryptoloop. Tato možnost by se měla zobrazit ihned poté, co zapnete podporu obecného loopback zařízení.
2Chcete-li zapnout šifrovací rozhraní API, přejděte z hlavní nabídky na položku „Crypto-graphic options“ (Možnosti šifrování). Zde můžete bez rizika zapnout většinu algoritmů. Doporučuji zapnout následující možnosti:

```
-- Cryptographic API<*> HMAC support< > Null algorithms<*> MD4 digest algorithm<*> MD5 digest algorithm<*> SHA1 digest
     algorithm<*> SHA256 digest algorithm<*> SHA384 and SHA512 digest algorithms<*> DES and Triple DES EDE cipher
     algorithms<*> Blowfish cipher algorithm<*> Twofish cipher algorithm<*> Serpent cipher algorithm<*> AES cipher
     algorithms<*> CAST5 (CAST-128) cipher algorithm<*> CAST6 (CAST-256) cipher algorithm<*> Deflate compression
     algorithm< > Testing module
```

Jestliže se rozhodnete, že z těchto funkcí vytvoříte moduly, nezapomeňte nejdříve při spuš-tění systému načíst příslušné moduly (jmenují se cryptoloop, aes atd.).

5. Vytvořte jádro a moduly a nainstalujte je. Pokud například používáte zavaděč lilo v počítači x86, lze to provést takto:

   ```
   make
   make modules_install
   cp arch/i386/boot/bzImage /boot/kernel-2.6.1
   lilo
   ```

6. Načtěte požadované moduly při spuštění systému. V různých distribucích se to provádí odlišně. V systému Gentoo lze například tyto moduly přidat do souboru /etc/modu-les.autoload/kernel-2.6, jinde to může být třeba /etc/modprobe.preload. Jestliže jste přeložili Cryptoloop jako modul, je nutné jej načíst jako první. Automaticky také načte základní modul loopback zařízení. Modul můžete načíst ručně příkazem:
   ```
   modprobe cryptoloop
   ```

Získání uživatelských nástrojů

Ovladač Cryptoloop vyžaduje aktualizované uživatelské nástroje, aby mohl v praxi vytvořit a při-pojit šifrovaný systém souborů. Potřebujete k tomu aktualizovaný balíček util-linux, který získáte na adrese *http://ftp.cwi.nl/aeb/util-linux/util-linux-2.12.tar.gz*. Nejaktuálnější verze má číslo 2.12. Brzy budou k dispozici nové verze, které pravděpodobně přinesou zásadní změny. Než tedy upgradujete na novější verzi, nezapomeňte zkontrolovat tento dokument HOWTO. Bohužel se můžete setkat s mnoha různými záplatami balíčku util-linux. Liší se způsobem vytváření a připo-jování šifrovaných oddílů. Chcete-li použít balíček util-linux 2.12 s jádrem 2.6, musíte aplikovat alespoň následující dvě záplaty:

1Kombinovaná záplata losetup *(http://www.stwing.org/~sluskyb/util-linux/losetup -combined.patch*).
2Záplata util-linux 2.6 (*http://www.ece.cmu.edu/~rholzer/cryptoloop/util-linux-2.12-kernel-*

2.6.patch). Stáhněte si balíček util-linux a dvě výše uvedené záplaty. Nejdříve rozbalte balíček util-linux a potom aplikujte dvě záplaty:

tar xvfz util-linux-2.12.tar.gz cd util-linux-2.12 patch -p1 < /cesta_k_souboru_záplaty/losetup-combined.patch patch -p1 < /cesta_k_souboru_záplaty/util-linux-2.12-kernel-2.6.patch

Po použití záplat přeložte a nainstalujte balíček util-linux podle pokynů v souboru INSTALL. Doporučuji použít systém Gentoo Linux, který tyto záplaty aplikuje automaticky, když se záplaty balíčku util-linux objeví. V jiných distribucích mohou být také dostupné verze balíčku util-linux, které již mají tyto záplaty aplikovány.

Nastavení loopback zařízení

Cryptoloop lze použít na soubor nebo celým systém souborů. Následující popis se týká nastavení tohoto nástroje pro určitý oddíl. Může se jednat o libovolný vybraný oddíl. V následujícím příkla-du se používá /dev/sda1. Jako šifrovací algoritmus jsem zvolil AES, ale můžete jej nahradit za jakoukoli oblíbenou šifru, kterou jste v jádře zapnuli. Seznam algoritmů podporovaných aktuálně spuštěným jádrem můžete zkontrolovat v souboru /proc/crypto. Vynikajícím zdrojem informací o různých šifrovacích algoritmech jsou knihy Bruceho Schneiera Applied Cryptography a Practical Cryptography. Rozumnou volbu pravděpodobně představují algoritmy AES i Serpent. Na algorit-mus AES se zaměřilo hodně pokusů o kryptoanalýzu a zatím nebyla zjištěna žádná vážná slabá místa. Algoritmus Serpent nebyl analyzován tak rozsáhle, ale podle názoru odborníků je ještě o něco silnější než AES. Serpent je však také pomalejší než AES. Vyhněte se algoritmu DES, který je zároveň pomalý a slabý. Alternativou může být algoritmus Triple-DES, ale AES je pravděpo-dobně bezpečnější a rychlejší, takže již není důvod nadále Triple-DES používat.

1. Než na oddílu vytvoříte šifrovaný systém souborů, doporučuje se oddíl naformátovat a zaplnit jej náhodnými daty. Díky tomu bude pro útočníka obtížnější nalézt na šifrovaném oddílu určité vzory.

   Varování! Dávejte pozor, jakou zde pro svůj oddíl zadáte hodnotu (parametr of). Pokud uděláte chybu, můžete snadno přepsat nesprávný oddíl náhodnými nesmyslnými daty!
   Oddíl můžete zaplnit náhodnými daty takto:

   dd if=/dev/urandom of=/dev/sda1 bs=1M

   Někdy se zobrazí chybová zpráva, že zařízení je plné. Tuto zprávu můžete ignorovat.

1Vyberte šifru a délku klíče. Seznam šifer podporovaných svým jádrem naleznete v soubo-ru /proc/crypto. Doporučuji použít algoritmus AES s 256bitovým klíčem.
2Nastavte loopback zařízení. Slouží k tomu příkaz losetup z balíčku util-linux. Následující příkaz vytvoří šifrovaný systém souborů pomocí loopback zařízení číslo 0 a pomocí šifry AES s 256bitovým klíčem na zařízení /dev/sda1:

   losetup -e aes-256 /dev/loop0 /dev/sda1

   Příkaz vyzve k zadání hesla. Vyberte silné heslo a pokuste se jej zapamatovat, aniž byste si jej museli poznamenat na štítek přilepený k monitoru. Práce s nástrojem Cryptoloop má jednu velkou nevýhodu. Vzhledem k tomu, že hodnota hash odvozená z hesla slouží jako šifrovací klíč, nelze heslo později snadno změnit. Nejjednodušší způsob změny hesla spočívá ve vytvoření nového šifrovaného oddílu nebo souboru, do kterého přesunete všech-na data. Proto je nutné vybrat silné heslo hned na začátku. Algoritmus AES je sice nejspíš dostatečně silný, ale pokud si zvolíte slabé heslo, bezpečnost se vytrácí.

   Pokud příkaz losetup není úspěšný a zobrazí chybovou zprávu INVALID ARGUMENT, jedná se o chybu balíčku util-linux. Zkontrolujte, zda jste postupovali podle výše uvede-ných pokynů pro instalaci opravené verze balíčku util-linux. Starší a nezáplatované verze předávají délku klíče odlišným způsobem a nefungují s rozhraním Crypto API jádra 2.6.

4. Vytvořte systém souborů. Můžete zvolit libovolný systém souborů. Následující příkaz vytvoří systém souborů ext3 file s použitím loopback zařízení:

   mkfs.ext3 /dev/loop0

5. Připojte šifrovaný systém souborů. Nejdříve musíte vytvořit připojovací bod, jako např. /mnt/crypto:

   mkdir /mnt/crypto

   Potom lze systém souborů připojit. V této fázi musíte příkazu mount explicitně sdělit, které loopback zařízení má použít:

   mount -t ext3 /dev/loop0 /mnt/crypto

1Nyní si se svým šifrovaným systémem souborů můžete začít hrát, dokud vás to neomrzí.
2Odpojte systém souborů. Když jste systém souborů dostatečně vyzkoušeli, odpojte jej:

   umount /mnt/crypto

8. Odpojte loopback zařízení – stále je připojeno k příslušnému oddílu. Odpojení provede příkazem:

   losetup -d /dev/loop0

Připojení šifrovaného souborového systému

U všech operací se zařízením Cryptoloop je důležité, aby byly načteny všechny potřebné modu-ly. Musíte načíst alespoň modul Cryptoloop a moduly pro každou šifru pomocí modprobe. Pokud jsou funkce přeloženy přímo do jádra, není to nutné.
Chcete-li připojit šifrovaný systém souborů vytvořený výše, můžete použít standardní příkaz mount z balíčku util-linux:

mount -t ext3 /dev/sda1 /mnt/crypto/ -oencryption=aes-256

Zobrazí se výzva k zadání hesla a systém souborů bude připojen standardním způsobem. Z funk-ce šifrování vyplývá, že se jedná o systém souborů Cryptoloop. Proto automaticky zvolí dostupné loopback zařízení.

Po dokončení práce systém souborů odpojte příkazem:

umount /mnt/crypto

Do souboru /etc/fstab lze přidat následující řádek:/dev/sda1 /mnt/crypto ext3 noauto,encryption=aes-256 0 0

Nyní můžete zařízení jednoduše připojit příkazem:

mount /mnt/crypto

To je vše. Hodně štěstí.

Použití souboru místo oddílu

Stejně snadné je vytvořit šifrovaný systém souborů uvnitř souboru v jiném systému souborů. Tato možnost je užitečná zejména v případech, kdy chcete tento soubor zálohovat – například vypálením na disk DVD atd. Soubor lze pak také snadno přenést do jiných počítačů.

Chcete-li nejdříve vytvořit soubor velikosti 100 MB, který bude obsahovat náhodná data, zadejte následující příkaz:

dd if=/dev/urandom of=/mojedata.aes bs=1k count=100000

Pokud chcete velikost souboru změnit, upravte příslušným způsobem hodnotu parametru count. Výše uvedený příkaz vytvoří 100 000 bloků s velikostí 1 kB, ale tuto hodnotu můžete podle potře-by změnit. Musíte pouze zajistit, aby měl soubor dostatečnou velikost k vytvoření zvoleného syste-mu souborů. Místo parametru /mojedata.aes můžete zadat libovolný název souboru a cestu, pokud je na oddílu dostatek místa.

Pak lze v rámci tohoto souboru vytvořit šifrovaný systém souborů podobně jako ve výše uvedeném příkladu:

losetup -e aes-256 /dev/loop0 /mojedata.aes

Nyní je možné vytvořit systém souborů:

mkfs.ext3 /dev/loop0

a připojit jej:

mount -t ext3 /dev/loop0 /mnt/crypto

Nakonec loopback zařízení odpojte:

umount /mnt/crypto losetup -d /dev/loop0

Později můžete systém souborů připojit takto:

mount /mojedata.aes /mnt/crypto -oencryption=aes-256

Chcete-li soubor přesunout nebo jej vypálit na disk CD či DVD, nezapomeňte jej nejdříve *odpojit*.

Šifrovaný kořenový oddíl

Příprava systému

V tomto dokumentu naleznete informace, jak zabezpečit osobní data zašifrováním kořenového oddílu v systému Linux pomocí silné šifry. Své komentáře posílejte Christophu Devinovi (*http://www.cr0.net:8040/about/*).

Nastavení rozložení oddílů

Pevný disk (budeme používat označení hda) by měl obsahovat alespoň tři oddíly:

hda1: tento malý nešifrovaný oddíl požádá o zadání hesla, aby bylo možné připojit šifro-vaný kořenový oddíl

hda2: tento oddíl bude obsahovat šifrovaný kořenový oddíl. Ponechte mu dostatečnou velikost

hda3: na tomto oddílu se nachází současná instalace systému GNU/Linux

V této fázi nejsou oddíly hda1 ani hda2 využité. Na oddílu hda3 je aktuálně nainstalována použí-vaná distribuce systému Linux. Adresáře /usr a /boot *nelze* z tohoto oddílu přenést jinam. Následuje příklad možného rozložení oddílů:

```
# fdisk -l /dev/hda

Disk /dev/hda: 255 heads, 63 sectors, 2432 cylinders Units = cylinders of 16065 * 512 bytes

   Device Boot Start End Blocks Id System /dev/hda1 1 1 8001 83 Linux /dev/hda2 2 263 2104515 83
Linux /dev/hda3 264 525 2104515 83 Linux /dev/hda4 526 2047 12225465 83 Linux
```

Požadované balíčky

Pokud používáte distribuci Debian, jsou nutné následující balíčky:

```
apt-get install gcc make libncurses5-dev patch bzip2 wget
```

Chcete-li si usnadnit kopírování a vkládání, můžete také nainstalovat:

```
apt-get install lynx gpm
```

V ostatních distribucích budou balíčky podobných jmen.

Instalace systému Linux 2.4.29

Existují dva hlavní projekty, které do jádra doplňují podporu šifrování pomocí loopback zařízení: Cryptoloop a Loop-AES. Tento postup je založen na nástroji Loop-AES, informace o nástroji Cryp-toloop najdete v knize v návodu „Cryptoloop“. Nástroj Loop-AES nabízí mimořádně rychlou a vysoce optimalizovanou implementaci algoritmu Rijndael v jazyce Assembler. Díky tomu posky-tuje v počítačích s procesorem IA-32 (x86) maximální výkon. Kromě toho existují určité pochyb-nosti týkající se bezpečnosti nástroje Cryptoloop.

V některých distribucích mohou být nástroje Loop-AES již součástí jádra a připraveny k použití (vyzkoušejte make menuconfig, viz instrukce dále, a přesvědčte se). Jestliže vaše jádro tuto pod-poru nemá, budete si muset vytvořit potřebné zázemí sami. Nejdříve si stáhněte a rozbalte balíček Loop-AES:

```
cd /usr/src wget http://loop-aes.sourceforge.net/loop-AES/loop-AES-v3.0b.tar.bz2 tar -xvjf loop-AES-v3.0b.tar.bz2
```

Poté je nutné stáhnout a nainstalovat záplatu zdrojového kódu jádra:

```
wget http://ftp.kernel.org/pub/linux/kernel/v2.4/linux-2.4.29.tar.bz2 tar -xvjf linux-2.4.29.tar.bz2 cd linux-2.4.29 rm include/linux/loop.h
drivers/block/loop.c patch -Np1 -i ../loop-AES-v3.0b/kernel-2.4.28.diff
```

Nastavte mapu klávesnice:

```
dumpkeys | loadkeys -m -> drivers/char/defkeymap.c
```

Dále nakonfigurujte jádro. Nezapomeňte nastavit následující možnosti:

make menuconfig

Block devices --->

<*> Loopback device support
[*] AES encrypted loop device support (NEW)

<*> RAM disk support(4096) Default RAM disk size (NEW)[*] Initial RAM disk (initrd) support

File systems --->

<*> Ext3 journalling file system support
<*> Second extended fs support

(important note: do not enable /dev file system support)

Přeložte jádro a nainstalujte je:
make dep bzImage make modules modules_install cp arch/i386/boot/bzImage /boot/vmlinuz

Používáte-li zavaděč grub, aktualizujte soubor /boot/grub/menu.lst nebo /boot/grub/grub.conf:

cat > /boot/grub/menu.lst << EOF default 0 timeout 10 color green/black light-green/black title Linux
root (hd0,2) kernel /boot/vmlinuz ro root=/dev/hda3 EOF

Jinak aktualizujte soubor /etc/lilo.conf a spusťte zavaděč lilo:

cat > /etc/lilo.conf << EOF lba32 boot=/dev/hda prompt timeout=60 image=/boot/vmlinuz
label=Linux
read-only
root=/dev/hda3

EOF lilo

Nyní můžete systém restartovat.

Instalace systému Linux 2.6.10

Postupujte stejně jako v předchozí části, ale tentokrát použijte záplatu *kernel-2.6.10.diff* pro Loop-AES. Také zkontrolujte, zda *není* aktivována podpora zařízení cryptoloop. Uvědomte si, že pod-pora modulů vyžaduje, abyste nainstalovali balíček module-init-tools.

Instalace balíčku util-linux-2.12p

Chcete-li doplnit podporu silného šifrování, musíte záplatovat a znovu přeložit program losetup, který je součástí balíčku util-linux. Stáhněte si, rozbalte a záplatujte balíček util-linux:

cd /usr/src wget http://ftp.kernel.org/pub/linux/utils/util-linux/util-linux-2.12p.tar.bz2 tar -xvjf util-linux-2.12p.tar.bz2 cd util-linux-2.12p patch -Np1 -i ../loop-AES-v3.0b/util-linux-2.12p.diff

Pokud chcete používat hesla kratší než 20 znaků, zadejte:

CFLAGS="-O2 -DLOOP_PASSWORD_MIN_LENGTH=8"; export CFLAGS

V prvé řadě byste měli dbát na bezpečnost. Proto rozhodně nepovolujte hesla kratší než 20 znaků.Soukromí dat není zadarmo, ale je nutné za ně „platit" formou dlouhých hesel.Přeložte program losetup a nainstalujte jej jako uživatel root:
./configure && make lib mount mv -f /sbin/losetup /sbin/losetup~ rm -f /usr/share/man/man8/losetup.8* cd mount gzip losetup.8 cp losetup /sbin cp losetup.8.gz /usr/share/man/man8/ chattr +i /sbin/losetup

Vytvoření šifrovaného kořenového oddílu

Zaplňte cílový oddíl náhodnými daty:

shred -n 1 -v /dev/hda2

Nastavte šifrovací loopback zařízení:

losetup -e aes256 -S xxxxxx /dev/loop0 /dev/hda2

Jako prevence optimalizovaných slovníkových útoků se doporučuje přidat parametr -S xxxxxx, kde „xxxxxx" je náhodně zvolená

hodnota seed (můžete například zadat „gPk4lA“). Vybranou hodnotu seed si poznamenejte, abyste ji nezapomněli. Chcete-li se vyhnout problémům s mapou klávesnice při spouštění, nepoužívejte také v hesle znaky mimo základní část tabulky ASCII (např. písmena s diakritikou). Web Diceware (*http://www.diceware.com/*) umožňuje jednoduše vytvořit silná hesla, která si však můžete snadno zapamatovat.

Nyní vytvořte na zařízení souborový systém ext3:

```
mke2fs -j /dev/loop0
```

Zkontrolujte, zda jste zadali správné heslo:

```
losetup -d /dev/loop0 losetup -e aes256 -S xxxxxx /dev/loop0 /dev/hda2

mkdir /mnt/efs mount /dev/loop0 /mnt/efs
```

Můžete porovnat šifrovaná a nešifrovaná data:

```
xxd /dev/hda2 | less xxd /dev/loop0 | less
```

Nyní je čas nainstalovat šifrovaný systém Linux. Používáte-li distribuci GNU/Linuxu (jako např. Debian, Slackware, Gentoo, Mandriva, RedHat/Fedora, SuSE atd.), spusťte následující příkaz:

```
cp -avx / /mnt/efs
```

Jestliže máte k dispozici knihu Linux From Scratch, postupujte podle popisu v příručce s následující změnou:
Kapitola 6 – Instalace balíčku util-linux:
Aplikujte záplatu Loop-AES po rozbalení zdrojových souborů.

Kapitola 8 – Nastavení systému LFS jako spustitelného:
Přejděte na další kapitolu, „Nastavení spouštěcího zařízení“.

Nastavení spouštěcího zařízení

Vytvoření ramdisku

Nejdříve použijte na šifrovaném oddílu příkaz chroot a vytvořte připojovací bod spouštěcího zaří-zení:

```
chroot /mnt/efs mkdir /loader
```

Potom vytvořte počáteční disk ramdisk (initrd – init ramdisk), který budete později potřebovat:

```
cd dd if=/dev/zero of=initrd bs=1k count=4096 mke2fs -F initrd mkdir ramdisk mount -o loop initrd ramdisk
```

Pokud používáte nástroj grsecurity, může se zobrazit chybová zpráva „Permission denied“ (Oprávnění odepřeno). V tomto případě je nutné spustit příkaz mount mimo prostředí příkazu chroot. Vytvořte hierarchii systému souborů a zkopírujte do ní požadované soubory:

```
mkdir ramdisk/{bin,dev,lib,mnt,sbin} cp /bin/{bash,mount} ramdisk/bin/ ln -s bash ramdisk/bin/sh mknod -m 600 ramdisk/dev/console c 5 1 mknod -m 600 ramdisk/dev/hda2 b 3 2 mknod -m 600 ramdisk/dev/loop0 b 7 0 cp /lib/{ld-linux.so.2,libc.so.6,libdl.so.2} ramdisk/lib/ cp /lib/{libncurses.so.5,libtermcap.so.2} ramdisk/lib/ cp /sbin/{losetup,pivot_root} ramdisk/sbin/
```

Jestliže se zobrazí zpráva typu „/lib/libncurses.so.5: No such file or directory“ nebo „/lib/libtermcap.so.2: No such file or directory“ (Tento soubor nebo adresář neexistuje), nemusíte se znepokojovat. Shell bash vyžaduje pouze jednu z těchto dvou knihoven. Chcete-li zjistit, která z knihoven je skutečně nutná, můžete zadat:

```
ldd /bin/bash
```

Přeložte program sleep, který zabrání tomu, aby byla výzva k zadání hesla zahlcena zprávami jádra (např. o registraci jednotlivých zařízení USB).

```
cat > sleep.c << "EOF" #include <unistd.h> #include
<stdlib.h>

    int main( int argc, char *argv[] ) { if( argc == 2 )
    sleep( atoi( argv[1] ) );

    return( 0 ); } EOF

gcc -s sleep.c -o ramdisk/bin/sleep rm sleep.c
```

Vytvořte skript init:

```
cat > ramdisk/sbin/init << "EOF" #!/bin/sh
```

```
/bin/sleep 3

echo -n "Zadejte hodnotu seed: " read SEED

/sbin/losetup -e aes256 -S $SEED /dev/loop0 /dev/hda2 /bin/mount -r -n -t ext3 /dev/loop0 /mnt

while [ $? -ne 0 ]
do /sbin/losetup -d /dev/loop0 /sbin/losetup -e aes256 -S $SEED /dev/loop0 /dev/hda2 /bin/mount -r -n -t ext3 /dev/loop0 /mnt
done

cd /mnt /sbin/pivot_root . loader exec /usr/sbin/chroot . /sbin/init EOF

chmod 755 ramdisk/sbin/init
```

Odpojte loopback zařízení a zkomprimujte soubor initrd:

```
umount -d ramdisk rmdir ramdisk gzip initrd mv
initrd.gz /boot/
```

Spouštění z disku CD-ROM

Rozhodně doporučuji spouštět systém z média pouze pro čtení, jako např. ze spustitelného diskuCD-ROM.Stáhněte si a rozbalte balíček syslinux:

```
wget http://ftp.kernel.org/pub/linux/utils/boot/syslinux/syslinux-3.07.tar.bz2 tar -xvjf syslinux-3.07.tar.bz2
```

Nakonfigurujte program isolinux:

```
mkdir bootcd cp /boot/{vmlinuz,initrd.gz} syslinux-3.07/isolinux.bin bootcd echo “DEFAULT /vmlinuz initrd=initrd.gz ro root=/dev/ram0” \
    > bootcd/isolinux.cfg
```

Vytvořte obraz ISO spustitelného disku CD-ROM:

```
mkisofs -o bootcd.iso -b isolinux.bin -c boot.cat \ -no-emul-boot -boot-load-size 4 -boot-info-table \ -J -hide-rr-
        moved -R bootcd/ cdrecord -dev 0,0,0 -speed 4 -v bootcd.iso

rm -rf bootcd{,.iso}
```

Spouštění z oddílu pevného disku

Spouštěcí oddíl může přijít vhod, pokud ztratíte spustitelný disk CD. *Nezapomeňte, že oddíl hda1 umožňuje zápis, a proto není bezpečný. Používejte jej pouze v nouzových případech!*

Vytvořte na zařízení souborový systém ext2 a připojte jej:

```
dd if=/dev/zero of=/dev/hda1 bs=8192 mke2fs /dev/hda1 mount /dev/hda1 /loader
```

Zkopírujte jádro a počáteční disk ramdisk:

```
cp /boot/{vmlinuz,initrd.gz} /loader
```

Používáte-li zavaděč grub:

```
mkdir /loader/boot cp -av /boot/grub /loader/boot/ cat > /loader/boot/grub/menu.lst << EOF default 0 timeout 10 color green/black light-
green/black title Linux
    root (hd0,0)
    kernel /vmlinuz ro root=/dev/ram0
    initrd /initrd.gz

EOF grub-install --root-directory=/loader /dev/hda umount /loader
```

Jestliže používáte zavaděč lilo:

```
mkdir /loader/{boot,dev,etc} cp /boot/boot.b /loader/boot/ mknod -m 600 /loader/dev/hda b 3 0 mknod -m 600 /loader/dev/hda1 b 3 1 mknod -m
600 /loader/dev/hda2 b 3 2 mknod -m 600 /loader/dev/hda3 b 3 3 mknod -m 600 /loader/dev/hda4 b 3 4 mknod -m 600 /loader/dev/ram0 b 1 0 cat
> /loader/etc/lilo.conf << EOF lba32 boot=/dev/hda prompt timeout=60 image=/vmlinuz
    label=Linux
    initrd=/initrd.gz
    read-only
    root=/dev/ram0
```

EOF lilo -r /loader umount /loader

Závěrečné kroky

Stále v rámci prostředí příkazu chroot upravte soubor /etc/fstab tak, aby obsahoval řádek:

/dev/loop0 / ext3 defaults 0 1

Odstraňte soubor /etc/mtab a ukončete příkaz chroot (pomocí exit). Nakonec spusťte příkaz umount -d /mnt/efs a restartujte systém. I v případě nějaké chyby můžete stále spustit nešifro-vaný oddíl zadáním příkazu „Linux root=/dev/hda3“ na výzvu „LILO:“.

Pokud vše proběhlo úspěšně, můžete nyní změnit oddíly na disku a zašifrovat oddíly hda3 i hda4. V následujících skriptech se předpokládá, že na oddílu hda3 bude uloženo odkládací zařízení swap a oddíl hda4 bude obsahovat adresář /home. Nejdříve byste měli oba oddíly inicializovat:

shred -n 1 -v /dev/hda3 shred -n 1 -v /dev/hda4 losetup -e aes256 -S xxxxxx /dev/loop1 /dev/hda3 losetup -e aes256 -S xxxxxx /dev/loop2 /dev/hda4 mkswap /dev/loop1 mke2fs -j /dev/loop2

Dále vytvořte skript ve spouštěcím adresáři systému a aktualizujte soubor fstab:cat > /etc/init.d/loop << "EOF"#!/bin/sh

```
    if [ "`/usr/bin/md5sum /dev/hda1`" != \
            "5671cebdb3bed87c3b3c345f0101d016 /dev/hda1" ] then echo -n
            "VAROVÁNÍ! Ověření integrity oddílu hda1 se NEZDAŘILO -
            stiskněte klávesu Enter." read fi

echo "První heslo zvolené výše" | \ /sbin/losetup -p 0 -e aes256 -S xxxxxx /dev/loop1 /dev/hda3

echo "Druhé heslo zvolené výše" | \ /sbin/losetup -p 0 -e aes256 -S xxxxxx /dev/loop2 /dev/hda4

/sbin/swapon /dev/loop1

  for i in `seq 0 63` do echo -n -e "\33[10;10]\33[11;10]"
  > /dev/tty$i done

EOF

chmod 700 /etc/init.d/loop ln -s ../init.d/loop /etc/rcS.d/S00loop vi /etc/fstab ... /dev/loop2 /home ext3 defaults 0 2
```

Tisk v Linuxu

Úvod

Tento dokument by měl obsahovat všechno, co potřebujete vědět k nastavení tiskových služeb na linuxovém stroji. Představuje souhrn informací o tom, jak v Linuxu cokoliv generovat, prohlížet, faxovat a tisknout. Téměř vše platí i pro uživatele jiných volně šiřitelných operačních systémů typu Unix. Život tomu chtěl, že je to o něco složitější než v klikacím světě Microsoftu a Apple, na druhé straně je to ale o něco pružnější a určitě jednodušší na správu ve velkých sítích.

Dokument je strukturován tak, že většině lidí bude stačit přečíst jenom první polovinu, nebo tak nějak. Většina obskurních a na situaci závislých informací je uvedena v druhé polovině a lze je snadno najít v obsahu. Informace z kapitoly „Ghostscript“ nebo „Sítě“ jsou pravděpodobně užitečné téměř pro každého.

Shledáte-li tento dokument nebo internetové stránky *http://www.linuxprinting.org/* užitečnými,
uvažujte o nákupu něčeho (třeba inkoustu), čím byste tyto stránky a úsilí jejich tvůrců podpořili. Poslední verzi tohoto dokumentu najdete na *http://www.linuxprinting.org/*; také jej najdete jako součást LDP na *http://www.tldp.org/* a na nejbližších zrcadlech.

Terminologie

V tomto dokumentu se pokusím používat konzistentní terminologii tak, aby byla přínosem pro všechny uživatele volně šiřitelných systémů typu Unix, a dokonce i jiných volně šiřitelných syste-mů. Bohužel však existuje celá řada nejednoznačných názvů a také množství sice jednoznačných, avšak nevhodně zvolených názvů. Uveďme si kvůli srozumitelnosti krátký slovníček výrazů:

UNIX

UNIX je operační systém vyvinutý výzkumníky v Bellových laboratořích. Na jeho kódu je založena řada (často komerčních) operačních systémů, které také spadají do této kategorie.

Un*x

Un*x není příliš vhodně zvolený symbol. Používá se k označení operačních systémů typu Unix. Operační systém typu Unix poskytuje něco, co se podobá programovému rozhraní POSIX, resp. jeho přirozenému aplikačnímu rozhraní (API, Application Programming Interface). Do kategorie Un*x spadají Linux, FreeBSD, Solaris, AIX, a dokonce i některé specializované systémy, např. Lynx nebo QNX.

Linux

Linux je jádro typu Unix doplněné různým periferním softwarem, které napsal Linus Torvalds a stovky dalších programátorů. Toto jádro je základem nejrozšířenějšího operačního systému kategorie Un*x.

GNU

Projekt GNU (*„GNU's Not Unix“*) dlouhodobě usiluje o vytvoření plně volně šiřitelného operační-ho systému typu Un*x. Tento projekt je v mnoha ohledech zdrojem nejmodernějších trendů v oblasti volně šiřitelného softwaru.

GNU/Linux

Operační systém GNU/Linux je kompletní systém složený z jádra a z periferních programů, který poskytuje prostředí GNU, v němž mohou běžet knihovny, utility, uživatelské programy atd. Pro-dejci kompletního systému GNU/Linux jsou firmy Red Hat, Debian, Caldera, SuSE, TurboLinux atd. GNU/Linux se ovšem velmi často zkracuje na pouhý Linux.

Přímo k věci

Nejrychlejší metoda, jak začít, bude pravděpodobně použít konfigurační nástroje dodávané s vaším systémem. Za předpokladu, že obsahují podporu vašeho ovladače a ovladač vaší tiskár-ny, mělo by být docela snadné takto rychle provést základní nastavení.

Informace o konfigurač-ních nástrojích různých dodavatelů najdete v kapitole „Řešení jednotlivých distribucí“.

Pokud konfigurační nástroje dodavatele nefungují, měli byste zjistit, zda vaše tiskárna vůbec fun-govat *může*. V kapitole „Seznam tiskáren z hlediska kompatibility s Linuxem“ najdete odkaz na seznam kompatibilních tiskáren.

Pokud tiskárna má s určitým ovladačem fungovat, zjistěte, zda jej máte nainstalován, a pokud ne, nainstalujte jej. Typicky byste měli být schopni najít upravený balík Ghostscript obsahující nověj-ší verzi tohoto programu a doplňující ovladače. Pokud jej nenajdete, můžete si jej přeložit sami
– tento proces není triviální, ale je dobře dokumentovaný. Podrobnosti o systému Ghostscript
najdete v kapitole „Ghostscript“.Po instalaci správného ovladače zkuste tiskárnu znovu nakonfigurovat pomocí nástrojů dodavate-le. Nepovede-li se to, zkuste jiný nástroj podle doporučení v kapitole „Jak to funguje“. Pokudnepomůže ani to, budete muset provést konfiguraci sami, opět podle návodu v téže kapitole.

Pokud tiskárna stále nefunguje, bude potřeba něco diagnostiky. V takovém případě bude nejlepší si nejprve přečíst větší část tohoto dokumentu, abyste zjistili, jak věci mají fungovat. Pak budete mít výrazně lepší pozici na hledání potíží.

Kde naleznete pomoc

Diskusní skupiny comp.os.linux.hardware, comp.os.linux.setup a comp.periphs.printers mají urči-tou část věnovanou otázkám souvisejícím s obecnou problematikou tisku. Existují i další hojně navštěvované diskusní skupiny; najdete je mimo jiné i v archivu Googlu. Dále jsou na Internetu diskusní skupiny linuxprinting.foo dostupné jak běžnými prohlížeči, tak i pomocí protokolu NNTP; zkuste si je najít.
Také zkuste sami pohledat na příbuzných stránkách na Internetu. Vhodným místem, kde můžete
začít, je například *http://www.linuxprinting.org/,* kde je mimo jiné i řada relevantních odkazů. Další pomoc naleznete v různých diskusních skupinách, e-mailových konferencích, zákaznických linkách distributora atd. Chcete-li kontaktovat přímo mě, učiňte tak, prosím, na diskusním fóru *http://www.linuxprinting.org/*, což poskytne příležitost k odpovědi i jiným účastníkům a pomůže i jiným nešťastníkům s podobnými problémy.

Jak tisknout

Tisknout budete různými příkazy podle toho, jaký tiskový spooler máte nainstalován.

Pomocí LPD a příkazu lpr v systému BSD

Pokud jste si pro tisk již nastavili lpd (nebo to udělal správce systému, případně už prodejce), stačí se už jen naučit používat příkaz lpr. Jeho popis a popis několika dalších příkazů souvisejících s tiskem, které byste pravděpodobně měli znát, naleznete v dokumentu Printing Usage HOWTO (*http://www.tldp.org/HOWTO/Printing-Usage-HOWTO.html*) anebo v manuálových stránkách lpr(1).

Stručně: Jméno fronty zadáte pomocí -P a pak zadáte jméno souboru, který má být vytištěn. Neza-dáte-li nic, tiskne se ze stdin. Volby ovladače se obvykle z lpr nezadávají, avšak některé systémy akceptují volby např. -o, -Z nebo -J.

Příklad 1. lpr

lpr /etc/hosts lpr -J “my hosts file” /etc/hosts lpr -P mylaserjet /etc/services

Pomocí LPD a příkazu lp v systému System V

V různých variantách Unixu se můžete setkat se dvěma množinami příkazů. Tiskové systémy LPD v operačních systémech založených na BSD (*BSD, Linux) používají lpr (tisk), lpq (vypiš frontu), lprm (zruš úlohu), zatímco systémy na bázi System V používají lp (tisk), lpstat (vypiš frontu), cancel (zruš úlohu). Jsou to například systémy Solaris, SCO a řada dalších.

Konvence a příkazy SYSV jsou pochopitelně popsány v manuálových stránkách příkazu lp. Fron-tu zadáte volbou -d a dále volitelně jméno souboru. Není-li zadáno, bude se vypisovat ze stdin.

Příklad 2. lp

lp /etc/hosts lp -d mylaserjet /etc/services

Pomocí CUPS

CUPS nabízí rozhraní pro příkazové řádky jak System V, tak i systémů z Berkeley. Znamená to, že k tisku můžete používat jak příkaz lpr, tak i lp. Máte-li například balík skriptů, které obsahují výlučně lp, nebo jste zvyklí jenom na jeden ze systémů z kuchyně System V, resp. BSD, je to znač-ná úleva.

Grafické tiskové nástroje

Většina tiskových systémů sama o sobě nabízí jen velmi jednoduché řádkové rozhraní. Namísto přímého použití příkazu lpr dáte možná přednost nějakému grafickému rozhraní. Ta obecně umožňují nastavit různé možnosti tisku (tiskárnu, typ papíru, řazení, počet kopií a podobně) snad-no použitelnou grafickou metodou. Některá z nich nabízejí i další funkce.

KDEPrint

KDEPrint zajišťuje uživatelský přístup k tiskovým subsystémům (CUPS, LPD, RLPR, LPRng atd.) pomocí grafického uživatelského rozhraní KDE. Pomocí tohoto nástroje můžete snadno tisknout, spravovat úlohy a tiskárny a taktéž tiskové démony. KDEPrint je náhradou starších nástrojů QtCUPS a KUPS. Je vhodný jak pro uživatele, tak i pro vývojáře. Od verze KDE 2.2.0 je jeho sou-částí a má řadu zajímavých funkcí.

KDEPrint má tiskové dialogové okno programu kprinter, které umožňuje výběr cílové tiskárny a změnu voleb. K cílovým tiskárnám patří i několik virtuálních tiskáren, kam můžete posílat elek-tronickou poštu, faxy nebo soubory PDF.

Program kprinter můžete používat s aplikacemi, které mají konfigurovatelný příkaz pro tisk. Jsou to například Mozilla nebo OpenOffice.

V dialogovém okně KDEPrint také můžete zadat Print Preview. Tento příkaz je realizován tak, že předá tiskový soubor přes různé filtry, které jej upraví pro tisk na obrazovce pomocí KGhostView nebo pomocí nějaké externí aplikace (např. gv).

Pomocí KJobViewer můžete prohlížet, přesouvat a rušit tiskové úlohy.
Další informace o KDEPrint naleznete na *http://printing.kde.org/*.

XPP

Pokud jako tiskový spooler používáte CUPS, můžete použít program XPP (viz obrázek 11.4).
Generuje se pomocí knihovny FLTK, takže je nezávislý na pracovním prostředí. Chcete-li tímto programem tisknout, spusťte xpp a vyberte soubor (nebo jej nevybírejte, pokud chcete xpp použít namísto lpr pro tisk ze stdin). Pak ze seznamu nakonfigurovaných tiskáren zvolte tiskárnu a případné parametry, které chcete pro tisk použít. Na obrázku 11.5 vidíte příklad panelu s parametry se zvýrazněním standardních parametrů CUPS.

Pokud jej používáte se systémem rozhraní ovladačů Foomatic, můžete nastavovat i číselné parametry, které CUPS normálně nepodporuje. Obvykle se týkají takových funkcí jako barevné ladění, nastavení kazet atd. Příklad viz obrázek 11.6.

Všechny volby a vybranou tiskárnu lze uchovat pomocí tlačítka „Save Settings“.

GPR

GPR (*http://www.compumetric.com/linux.html*, autor Thomas Hubbell) využívá k filtrování post-skriptových úloh kód z CUPS a poskytuje možnost snadného řízení úloh. Některé volby (např. vícecestné tisky, výběr stránky apod.) jsou implementovány přímo pomocí GPR, zatímco většina ostatních až tiskárnou, resp. filtrem spooleru.

GPR pracuje s LPD nebo s LPRng nebo může být přeloženo zvlášť pro použití s GNUlpr. Je-li pře-loženo normálně, používá přímo libppd (firmy VA) k vytváření postskriptu pro určitou tiskárnu, který předá příkazu lpr. Je-li přeloženo pro GNU variantu lpr, předá příkazu lpr společně s uve-denými volbami postskriptovou úlohu v původní (nemodifikované) podobě. To je pravděpodob-ně lepší způsob, neboť umožňuje spooleru přesměrování postskriptu na různé tiskárny, je-li to nutné; bohužel vyžaduje GNU lpr, který se příliš nepoužívá (i když jeho instalace je triviální). Při používání GPR nejdříve vyberte tiskárnu (jménem fronty LPD) a přesvědčte se, že GPR zaved-lo správný soubor PPD. Pokud se tak nestalo, musíte zadat jméno souboru PPD a v konfigurač-ním dialogu zadat volby pro tiskárnu (dialog spustíte tlačítkem Printer Configuration; obsahuje různá nastavení tiskových voleb definovaných PPD).

Jakmile zkonfigurujete tiskárnu v GPR, můžete tisknout úlohy tak, že zadáte jméno souboru a vyberete příslušné volby z panelů „Common“ a „Advanced“. Volby z „Common“ implementuje pro všechny tiskárny přímo GPR, zatímco volby „Advanced“ pro danou tiskárnu jsou definovány

v souboru PPD. Panely s volbami jsou na obrázcích 8 a 9. Novější a modernější náhradou GPR se zdá být GtkLP (*http://gtklp.sourceforge.net/*), viz následující obrázek.

Tisková zařízení jádra

Pro obsluhu paralelního portu existují dvě úplně odlišná zařízení – které z nich používáte, závisí na verzi vašeho jádra (a to můžete zjistit příkazem uname -a). Ovladače se změnily v jádře 2.1.33; prakticky všechny dnešní systémy používají jádra 2.2 nebo vyšší; takže klidně můžete přeskočit na kapitolu popisující zařízení parport.

Poznámka platná pouze pro oba typy ovladačů: Nejdůležitější asi je, že řada uživatelů zjistila, že Linux nedetekuje paralelní port, pokud v BIOSu nevypnou podporu „Plug-and-Play“. (Není to velké překvapení, PnP záznamy pro jiná než PCI zařízení,

používané ve Windows a obdobných systémech, představují takovou malinkou katastrofu.)

Zařízení lp (jádra <= 2.1.32)

Za předpokladu, že máte přeloženou nebo nahranou podporu zařízení lp (pokud je nahrána, bude výstup příkazu cat /proc/devices obsahovat zařízení lp), pak jádro (<= 2.1.32) poskytuje jedno nebo více zařízení /dev/lp0, /dev/lp1, /dev/lp2. Tato zařízení se *nepřiřazují* dynamic-ky, každé odpovídá konkrétní hardwarové vstupně-výstupní adrese. Znamená to, že první a jedi-ná tiskárna může být klidně lp0 nebo lp1 v závislosti na hardwaru. Prostě zkuste obojí.

Někteří uživatelé si stěžují, že obousměrný port není detekován, pokud použijí starší jednosměr

ný kabel. V takovém případě zkontrolujte, zda používáte správný kabel. Na jednom portu nemůžete současně spustit ovladače (moduly) plip a lp (s jádrem 2.0). Může-te nicméně kdykoliv jeden nebo druhý ovladač nahrát ručně, případně démonem *kerneld* u jader

2.x (a novějších 1.3.x). Při správném nastavení přerušení a dalších parametrů ovšem můžete na jednom portu používat plip a na druhém lp. Úspěšně se to podařilo úpravami ovladačů; zajíma-lo by mě, zda se to někomu povedlo pouze vhodnými parametry příkazů.

Existuje nástroj tunelp, který s jádrem 2.0 umožňuje nastavovat přerušení a další parametry para-

lelních portů.Pokud je ovladač lp součástí jádra, přijímá jádro parametr lp= pro nastavení přerušení a vstupně--výstupní adresy.

Pokud je ovladač lp součástí jádra, můžete použít příkazový řádek programů LILO/LOADLIN a nastavit adresy a přerušení, které bude ovladač používat. Syntaxe: lp=port0[,irq0[,port1[,irq1[,port2[,irq2]]]]]. Například: lp=0x378,0 nebo lp=0x278,5,0x378,7. Pokud chcete tuto funkci použít, musíte definovat *všechny* používané porty, neexistují žádné implicitní hodnoty. Vestavěný ovladač můžete vypnout parametrem lp=0.

Pokud ovladač nahráváte jako modul, můžete vstupně-výstupní adresu a přerušení definovat na příkazovém řádku insmod (nebo v souboru /etc/conf.modules, pokud používáte kerneld) s obvyklou syntaxí parametrů. Parametry jsou io=port0,port1, port2 a irq=irq0,irq1,irq2. Další informace viz manuálová stránka příkazu insmod.

Zdrojový kód ovladače paralelního portu v jádře 2.0 najdete v souboru /usr/src/linux/dri-vers/char/lp.c.

Zařízení parport (jádra >= 2.1.33)

Počínaje jádrem 2.1.33 (se záplatou až pro 2.0.30) je zařízení lp pouhým klientem nového zaříze-ní (ovladače/modulu) parport. Přidáním zařízení parport se odstranila řada problémů s původ-ním ovladačem zařízení lp – bylo možné sdílet port s jinými ovladači, bylo zavedeno dynamické přiřazování dostupných portů číslům zařízení namísto pevného vztahu mezi vstupně-výstupní adresou a zařízením a další.

Díky novému ovladači parport se mohla objevit celá řada nových paralelních ovladačů pro zaří-zení, jako jsou mechaniky Zip, CD-ROM a disky Backpack a další. Některé z nich existují i v jád-rech 2.0; zkuste je najít na webu.

Hlavním rozdílem, kterého si můžete všimnout ve vztahu k tisku, je dynamické přiřazení zařízení lpparalelním portům. Tedy to, co v Linuxu 2.0 bylo lp1, může být v Linuxu 2.2 klidně lp0. Pokud přecházíte z jádra s ovladačem lp na jádro s ovladačem parport, nezapomeňte to zkontrolovat.

Většina oblíbených problémů s tímto zařízením vzniká právě díky chybné konfiguraci:

Distribuce

Některé distribuce Linuxu nemají správně nastaven soubor /etc/modules.conf (nebo /etc/conf.modules), takže se ovladač v okamžiku potřeby nenahraje správně. S posledními ver-zemi nástroje modutils vypadá správný řádek v souboru modules.conf takto:

alias /dev/printers lp # jenom pro devfs? alias /dev/lp* lp # jenom pro devfs? alias parport_lowlevel parport_pc # chybí v Red Hat 6.0-6.1

BIOS

Řada BIOSů obsluhuje paralelní port jako zařízení Plug-and-Play. Tím se ovšem perfektně jednoduché a prakticky vždy přítomné zařízení rozšiřuje o naprosto zbytečné složitosti. Pokud Linux nedetekuje paralelní port správně, vypněte u něj podporu PnP (ve většině BIOSů nastavujete LPT1). Správné nastavení bude obvykle legacy, ISA nebo 0x378 – určitě ale ne disabled.

Doporučujeme také dokumentaci ve zdrojových kódech jádra (*http://people.redhat.com-/twaugh/parport/html/parportguide.html*) nebo stránky *http://people.redhat.com/twaugh/parport/*.

Sériová zařízení

Sériová zařízení se v Linuxu obvykle jmenují podobně jako /dev/ttyS1. Nástrojem stty můžete interaktivně zobrazovat nebo měnit nastavení sériových portů, nástrojem setserial pak můžete nastavovat některé další parametry, přerušení a vstupně-výstupní adresy nestandardních portů. Podrobnější debatu na téma sériových portů v Linuxu najdete v dokumentu Serial-HOWTO.

Pokud používáte pomalou sériovou tiskárnu s řízením toku, může dojít k tomu, že se ztrácejí části tiskových úloh. Může to způsobovat sériový port, který se standardně chová tak, že všechny nepřenesené znaky ze svého buferu odstraní 30 sekund po

zavření zařízení. Bufer má délku 4 096 znaků, a pokud tiskárna používá softwarové řízení toku a je dost pomalá na to, aby nestihla zpra-covat celý bufer do 30 sekund od chvíle, kdy tiskový program zavře sériový port, konec buferu se ztratí. Pokud příkazem cat soubor > /dev/ttyS2 správně vytisknete krátké soubory, ale u dlouhých se konec souboru ztratí, může jít o tento případ.

Interval 30 sekund je možné změnit parametrem closing_wait příkazu setserial (verze 2.12 a vyšší). Sériové porty se typicky inicializují voláním setserial v souboru rc.serial. Můžete toto volání upravit taky, aby kromě ostatních parametrů rovnou nastavilo i delší interval closing_wait.

USB zařízení USB 1.1

Podpora USB je v Linuxu značná. USB lze použít se všemi pozdními modely jádra 2.2 a se všemi jádry 2.4 a novějšími. Jádro musí pochopitelně mít zabudovanou podporu USB, buď jako součást jádra nebo jako zaváděný modul (doporučeno).

Modulární jádro vyžaduje tyto moduly:

usb-core.o
usb-uhci.o nebo uhci.o nebo usb-ohci.o
printer.o

Konkrétní verze modulů usb-uhci.o, uhci.o a usb-ohci.o závisí na typu základní desky, resp. adaptéru. Intel, Via a adaptéry založené na Via jsou UHCI (v tomto případě můžete použít buď usb-uhci.o nebo uhci.o). Typ HCI (Host Controller Interface) zjistíte příkazem lspci -v|grep HCI.

USB 2.0

Vysokorychlostní přenos na zařízení USB 2.0 vyžaduje připojení řadiče USB 2.0 a ovladač EHCI (ehci-hcd.o). Pro USB 2.0 je doporučeno jádro 2.4 nebo vyšší.

Praktické rady

Je nutno si zapamatovat, že zařízení USB jsou přidělována dynamicky. Tiskárna USB je při zapnu-tí nebo připojení přiřazena souboru zařízení (/dev/usb/lp*). V důsledku toho mohou být tisko-vé úlohy posílány na nesprávnou tiskárnu, neboť je zapínáte v určitém pořadí. Zasílání úloh na správnou fyzickou tiskárnu zajišťuje CUPS dle speciálního URI obsahujícího značku, model a séri-ové číslo tiskárny.

I když v Linuxu lze využívat většinu tiskáren USB, existují výjimky. Například zařízení MF od Epsonu (Stylus CX3200/CX5200) vracejí nesmysly, když prostřednictvím IOCTL zvolíte řetězec IEEE-1284 ID, například s kódem CUPS „usb“. Je nutno zvolit ID řetězec v souladu s proprietární metodou firmy Epson.

Získat ID řetězce z tiskáren USB můžete pomocí několika nástrojů, které napsal Till Kamppeter. Programy getusbprinterid.pl (*http://www.linuxprinting.org/download/printing/getusbprinterid.pl*) a usb_id_test.c (*http://www.linuxprinting.org/download/printing/usb_id/test.c*) jsou funkčně totož-né, první z nich je však napsaný v Perlu a druhý v C. Jak jsme se zmínili shora, nová zařízení MF od firmy Epson jsou výjimkou, avšak v nástroji ttink z balíku MTink (*http://xwtools.automatix.de/*) je proprietární metoda firmy Epson implementována.

Další informace o USB naleznete na Linux USB Website (*http://www.linux-usb.org/*).

Podporované tiskárny

Jádro Linuxu vám umožní bavit se s jakoukoliv tiskárnou, kterou připojíte k sériovému, paralelní-mu nebo USB portu, plus s jakoukoliv síťovou tiskárnou. To ale samo o sobě nestačí – musíte být také schopni generovat data, kterým bude tiskárna rozumět. Typicky mezi nepoužitelné tiskárny patří tiskárny označované jako „Windows“ nebo „GDI“. Jazyk pro ovládání těchto tiskáren je úplně nebo částečně nedokumentován, stejně jako podrobnosti o návrhu tiskového mechanismu. Typic-ky výrobce dodává s tiskárnou ovladač pro Windows a vesele je prodává pouze uživatelům Win-dows; proto se tyto tiskárny označují také jako *Winprinters*. V některých případech dodává výrob-ce ovladače i pro NT, OS/2 a jiné operační systémy.

Řada těchto tiskáren v Linuxu nefunguje. Některé z nich fungují, některé fungují pouze částečně (obvykle díky tomu, že někdo pomocí reverzního překladu ovladače zjistil, jak s tiskárnou zachá-zet). Viz seznam použitelných tiskáren dále v této kapitole.

Některé tiskárny jsou něco mezi. Například některé modely tiskáren NEC implementují zjednodušenou verzi ovládacího jazyka PCL, díky čemuž mohou v rozlišení 300 DPI na tiskárně tisknout programy ovládající PCL. Ovšem pouze NEC ví, jak z tiskárny dostat plné možné rozlišení 600 DPI.

Pokud už některou z těchto windowsových tiskáren máte, existují různé prapodivné cestičky, jak je využít i v Linuxu, jde ale o poměrně nehezké postupy. Další podrobnosti o využití těchto tis-káren najdete v kapitole „Tiskárny výlučně pro Windows“.

PostScript

Co se týče tiskáren v Linuxu, nejlepší volba je koupit tiskárnu, jejíž firmware nativně podporuje PostScript. Prakticky všechny unixové programy produkující tiskový výstup jej vytvářejí jako PostScript, takže je zjevně příjemné mít tiskárnu, která jej přímo podporuje. Bohužel, podpora PostSc
riptu bývá kromě laserových tiskáren jen zřídkavá, a někdy se prodává pouze jako drahý doplněk. Unixové programy a obecně publikační průmysl se ustálily na používání PostScriptu coby stan-dardního jazyka pro práci s tiskárnami. Má to několik důvodů:

Načasování

PostScript se objevil jako součást Apple Laserwriter, perfektní tiskárny k Macintoshům, které jsou zodpovědné za revoluci v publikování v 80. letech.

Nezávislost na zařízení

Pomocí PostScriptu je možné generovat výstup na rastrové obrazovce, vektorové obrazovce, faxu nebo prakticky jakémkoliv tiskovém zařízení bez nutnosti úprav původního programu. Výstup PostScriptu bude na jakémkoliv zařízení vypadat stejně, samozřejmě v rámci možností daného zaří-zení. Před vznikem PDF se k výměně složitějších dokumentů používal právě PostScript. Hlavním důvodem, proč tento standard nezůstal jediným, je to, že Windows typicky neobsahují prohlížeč PostScriptu, takže Adobe doplnil PostScript o hypertextové odkazy a kompresi, výsledek pojme-noval PDF, začal distribuovat prohlížeče tohoto formátu a vytvořil trh pro své „destilační" nástro-je (jejichž funkce plní i programy ps2pdf a pdf2ps balíku ghostscript).

Jde o kompletní programovací jazyk

Můžete pomocí něj vytvářet programy, které budou dělat prakticky cokoliv. Většinou je to užiteč-né k definici procedur, které pak v rámci celého dokumentu stále dokola opakují stejné akce, například vykreslení loga nebo pozadí stránek. Neexistuje ale žádný důvod, proč byste pomocí PostScriptu nemohli třeba počítat číslo π.

Jde o otevřený formát

PostScript je kompletně dokumentován ve volně dostupných publikacích (které najdete v každém slušnějším obchodě ve světě) a také na adrese *http://partners.adobe.com/asn/developer/techno-tes/postscript.html.* I když tvůrcem formátu a dominantní komerční implementace je Adobe, nezá-vislé implementace nabízejí i jiní výrobci, například Aladdin (tvůrce nástroje Ghostscript).

Tiskárny bez PostScriptu

Pokud nemáte finance na nákup (obvykle drahé) postscriptové tiskárny, můžete použít jakoukoliv tiskárnu, kterou podporuje Ghostscript, volně šířený interpret PostScriptu, používaný namísto přímé podpory PostScriptu tiskárnou. Většina linuxových distribucí obsahuje z licenčních důvodů pouze poněkud zastaralou verzi Ghostscriptu. Naštěstí obvykle existuje možnost získat i aktuální verzi Ghostscriptu.

Společnost Adobe vytvořila nový jazyk pro práci s tiskárnami, pojmenovaný *PrintGear*. Pokud vím, jde o výrazně zjednodušený binární jazyk s jistými rysy PostScriptu, nicméně nekompatibilní s PostScriptem. Neslyšel jsem, že by jej Ghostscript podporoval. Některé tiskárny s jazykem Print-Gear však podporují i jiné jazyky, jako například PCL, a tyto tiskárny pak v Linuxu bez potíží fun-gují (pokud je ovšem PCL implementován přímo v tiskárně, a ne v ovladači pro Windows).

Adobe dále nabízí implementaci PostSciptu, pojmenovanou *PressReady*. Chová se podobně jako Ghostscript a nabízí podporu PostScriptu na tiskárnách, které ji nativně neobsahují, nevýhodou ovšem je, že tento program funguje pouze ve Windows.

Které tiskárny fungují?

Chystáte-li se kupovat tiskárnu, máte několik možností, kde zjistit, zda bude fungovat. Kolektivněudržovaná databáze tiskáren (*http://www.linuxprinting.org/database.html*) se snaží o vyčerpávají-cí seznam stavu podpory různých tiskáren v Linuxu. Najdete zde i informace o tom, jaký ovladačs kterou tiskárnou použít.

Nejlepším tipem pro ty, kdo kupují novou tiskárnu, je seznam na adrese *http://www.linuxprinting.org/suggested.html.* Obsahuje zejména barevné inkoustové a černobílé laserové tis-kárny. Stránky dokonce můžete sponzorovat tím, že si koupíte tiskárnu od některého z autorizo-vaných prodejců uvedených na adrese *http://www.linuxprinting.org/affiliate.html.*

Některé fungující tiskárny a odkazy na další stránky naleznete na ghostscriptovém seznamu

*http://www.cs.wisc.edu/~ghost/doc/printer.htm.*Googlovské diskusní skupiny *http://groups.google.com/* obsahují stovky dobrozdání „funguje"a „nefunguje". Vyzkoušejte všechny tři, a až si vyberete, ověřte si, zda ji naleznete v databázi tis-káren *http://www.linuxprinting.org/database.html*, aby do budoucna byla v seznam uvedenasprávně.

Tiskárny z hlediska kompatibility s Linuxem

V této kapitola byla původně shrnuta databáze tiskáren *(http://www.linux-foundation.org/en/ OpenPrinting/Database/DatabaseIntro*) se specifikacemi zařízení, poznámkami, informacemi o ovladačích apod. Tuto databázi jsme z důvodu neaktuálnosti z knihy vyřadili a odkazujeme vás na výše uvedenou on-line databázi. On-line seznam je

interaktivní; do seznamu může přidávat položky každý a také se to průběžně děje, takže si seznam také pečlivě prohlédněte. Když v něm svoji tiskárnu nenaleznete, můžete ji do něj přidat.

Poznamenejme ještě, že tento seznam pochopitelně není úplně přesný; občas se v něm objeví nesprávné informace, které jsou později odstraněny. Neověřené položky jsou označeny hvězdič-kou (*). Než tiskárnu vybranou dle tohoto seznamu zakoupíte, pro jistotu si ještě ověřte její vlast-nosti v některé googlovské diskusní skupině.

Tiskárny jsou z hlediska kompatibility rozděleny do čtyř kategorií:

Dokonalá

Dokonalé tiskárny tisknou dokonale – lze využít všech možností tiskárny včetně barev, plné roz-lišovací schopnosti atd. V některých případech jsou jako dokonalé označeny tiskárny s nezdoku-mentovanými režimy „rozšířeného rozlišení“, které nefunguje; obecně je rozdíl v kvalitě tak malý, že nemá smysl se jím zabývat.

Dobrá

Tiskne dobře, ale má určité nedostatky buď při tisku nebo při jiných funkcích.

Částečně použitelná

Je možno tisknout, avšak pravděpodobně nikoli barevně anebo se špatným rozlišením. Informa-ce o nedostatcích viz on-line seznam.

Těžítko

Nic nevytisknete pořádně; obvykle kvůli nedostatkům ovladače a/nebo dokumentace, podle níž je možno ovladač napsat. Těžítko se někdy „zlepší“, a to buď když někdo přijde na to, že ovladač přece jen funguje, nebo když někdo napíše nový. Ani s jedním, ani s druhým však nepočítejte.

Vzhledem k tomu, že tyto informace pocházejí od tuctů různých osob, žádná z nich není zaruče-ná; položky označené hvězdičkou (*) jsou mírně podezřelé. Skutečnost si lze snadno ověřit na internetových stránkách daného ovladače nebo výrobce tiskárny.

Jak kupovat tiskárnu

Volba tiskárny není dnes nic jednoduchého, existuje celá řada modelů, z nichž je možno vybírat. Uveďme si několik tipů, podle nichž se při koupi můžete orientovat:

Cena

Dostanete to, za co zaplatíte. Většina tiskáren v ceně mezi 200–300 dolary tiskne v rozumné kva-litě, nicméně má vysoké náklady na stránku. U některých tiskáren stojí jedna či dvě nové náplně stejně jako celá tiskárna! Navíc nejlevnější tiskárny nevydrží příliš dlouho. Typicky mají nejlevněj-ší tiskárny střední dobu mezi poruchami zhruba tři měsíce – což je docela málo, pokud má být tiskárna hodně používána.

Inkoustové tiskárny

Tiskové hlavy inkoustových tiskáren se po jisté době nenávratně ucpou, čili nutná je možnost časem hlavu vyměnit. Tiskové hlavy těchto tiskáren jsou drahé, přičemž kombinovaná hlava s náplní stojí zhruba 10krát (!!!) tolik co samotná náplň. Proto je rozumné volit takové tiskárny, kde hlava není integrovaná s náplní. Tento typ tiskáren vyrábí Epson, HP u řady DeskJet používá kombinované hlavy. Canon používá tři nezávisle na sobě vyměnitelné náplně – což je asi nejlep-ší řešení. Na druhé straně, náplně do tiskáren HP nejsou až tak hrozně drahé a kvalita tisku těch-to tiskáren je asi nejlepší. Tiskárny Canon stojí ohledně kvality až na třetím místě – a tiskárny HP a Epson jsou zatím v Linuxu nejlépe podporované.

Laserové tiskárny

U laserových tiskáren je nutné měnit tiskový válec a toner plus zásobník nespotřebovaného tone-ru. Nejlevnější řešení kombinuje válec s tonerem v jedné velké náplni, tyto tiskárny mají největší provozní náklady. Nejlepší velkokapacitní tiskárny používají samostatný toner nebo alespoň samo-statnou tonerovou kazetu a samostatný zásobník na odpad.

Fotografické tiskárny

Nejlepší barevný výstup ve fotografické kvalitě poskytují tiskárny s plynulým tónováním, které kombinují halogenidy stříbra a laserové osvícení a vyrábějí – překvapivě – opravdové fotografie! Tyto tiskárny ovšem stojí daleko více než tiskárny obyčejné, ale například Ofoto.com nabízí jed-norázové tisky fotografií za rozumné ceny. Výsledek je vynikající, ani nejlepší inkoustové tiskárny nemají šanci na srovnání.

Nejvyšší a za rozumnou cenu dosažitelnou kvalitu nabízejí (nebo spíš nabízely?) sublimační tiskárny, například série Alp (používají tepelný přenos pevného barviva nebo přímo sublimaci bar-viva), nebo fototiskárny Sony běžné spotřebitelské úrovně. Podpora těchto tiskáren v Linuxu je, bohužel, slabá (v jedné referenci, kterou jsem získal, se hovoří o pruzích a zrnitém obrazu) a ani pak není jisté, zda bude možné použít nejvyšší kvalitu sublimačního přenosu. O tiskárnách Sony nemám žádné informace.

Běžnější inkoustové tiskárny pro tisk fotografií používají šesti nebo sedmibarevný tisk (CMYKcm nebo CMYKcmy). Provoz těchto speciálních tiskáren je drahý. Buď vám dojde modrá a musíte vyměnit celou barevnou kazetu nebo je možné doplňovat barviva samostatně, ale náplň pak stojí majlant. Drahý je také speciální papír – nemůžete očekávat vysokou kvalitu při tisku na běžný kancelářský papír. Viz dále kapitola věnovaná tisku fotografií a popis nastavení barev v Ghostscriptu.

Novější barevné laserové tiskárny jsou už mnohem levnější, pro běžný tisk mohou být zajímavé. Stránka na barevné laserové tiskárně vyjde levněji než na inkoustové tiskárně. Na fotografie však ještě nejsou příliš vhodné. Je docela možné, že barevné laserové tiskárny už brzy nahradí ty otravné černobílé.

Rychlost

Rychlost je úměrná možnostem procesoru tiskárny, kapacitě připojení a ceně tiskárny. Nejrychlej-ší tiskárny jsou síťové postscriptové tiskárny s výkonným interním procesorem. Rychlost tisku běž-ných tiskáren bude zčásti ovlivněna rychlostí vykreslování Ghostcriptu, proto je vhodné použít rozumně výkonný počítač – zejména stránky se spoustou barev mají velké nároky na paměť tis-kového počítače. Pokud máte dostatek paměti, mělo by všechno fungovat bez potíží.

Formuláře

Pokud chcete tisknout formuláře s více kopiemi, potřebujete jehličkovou (úhozovou) tiskárnu. Řada společností stále vyrábí jehličkové tiskárny, které většinou emulují tradiční modely firmy Epson a fungují tak bez potíží.

Samolepky

Existují dvě podporované řady speciálních tiskáren na samolepky – Dymo-Costar a Seiko SLP. Ostatní tiskárny mohou a nemusí pracovat. Avery vyrábí samolepicí štítky různých velikostí ve for-mátu A4, takže můžete štítky tisknout i na normálních tiskárnách.

Plottery

K tisku na velké formáty se dnes obvykle používají obří inkoustové tiskárny, oblíbené jsou mode-ly firmy HP. Existují i menší inkoustové tiskárny vhodné pro tisk na formát A3. Většina těchto tis-káren používá jazyky RTL, HP-GL nebo HP-GL/2, což jsou všechno proprietární vektorové jazyky společnosti HP, výstup je typicky generován přímo aplikačními programy.

Spoolovací programy

Až donedávna měli uživatelé Linuxu jednoduchou volbu – všichni používali stejný starý lpd, pře-vzatý prakticky doslova z kódu Net-2 BSD. Dokonce i dnes většina dodavatelů tento program distribuuje. Nicméně situace se mění. Systémy SVR4 včetně Sun Solaris používají úplně jiné řeše-ní spooleru, postavené kolem programu lpsched.

V současné době lze vybírat z více dobrých systémů. Postupně je všechny popíšu, přečtěte si popi-sy a rozhodněte se sami. Nejjednodušší moderní systém s grafickým rozhraním je PDQ, hodí se jak pro normální domácí uživatele, tak i (v hybridním uspořádání pdq/lprng) pro větší prostředí. V komerčním nasazení, kde se vesměs používají síťové postscriptové tiskárny, je dobrou alterna-tivou rozhraní jako GPR s LPRng; přímo pracuje s možnostmi PPD a má o něco pěknější rozhra-ní. V jiných případech je dobrou volbou CUPS, má vynikající podporu postscriptových tiskáren a nabízí dále podporu IPP, webové rozhraní a další funkce.

CUPS

CUPS (*http://www.cups.org/*) se stal ve většině současných distribucí standardním tiskovým systé-mem. Čím se liší od ostatních tiskových systémů? Je implementací protokolu IPP (Internet Printing Protocol), což je nový standard, který má vyřešit nedostatky protokolu LPD. CUPS rovněž pod-poruje LPD, SMB a AppSocket (JetDirect) s omezenou funkcionalitou. Implementaci řídil Michael Sweet ze společnosti Easy Software Products; je šířený pod GPL. Oproti původnímu protokolu LPD má IPP celou řadu výhod:

Plánovač je internetový server HTTP 1.1 a poskytuje i internetové rozhraní.
Můžete se dotázat zařízení IPP, které volby a formáty dokumentů podporuje.
Řízení přístupu, které může omezovat tiskové úlohy, řízení prací a příkazy pro správu systému, které přicházejí nebo odcházejí na určené počítače a tiskárny. Přístup k CUPS můžete podobně jako v programu Apache řídit prostřednictvím direktiv Allow a Deny.
Podpora proxy (neboť IPP používá http).

- Podpora šifrování. Všichni hlavní prodejci operačních systémů i tiskáren v současnosti aktivně podporují IPP. IPP je standardním tiskovým protokolem ve Windows 2000 (musí být nainstalován IIS) a může být lepší volbou pro uživatele volně šiřitelného softwaru než proprietární protokol SMB. Nicméně, ve Win-dows 2000 automaticky stahovaný ovladač tiskárny spolupracuje pouze s SMB, nikoli s IPP. Pro správce s mnoha windowsovskými klienty to může být důvod, proč pro sdílení tiskárny SMB dáva-jí přednost systémům Samba nebo CUPS. CUPS má řadu velice dobrých vlastností včetně např. voleb pro kvalitu tisku; rozhraní pro Inter-

net, GUI a příkazové řádky; a filtrovací systém založený na mime typech se silnou podporou Post-Scriptu.Existuje několik

množin PPD, které můžete používat s CUPS:

Vestavěné

Implicitní instalace CUPS obsahuje generické PPD pro devítijehlové a dvacetičtyřjehlové mati-cové tiskárny Epson, tiskárny Stylus Color a Stylus Photo firmy Epson, LaserJet a DeskJet firmyHP a tiskárny štítků Dymo. Umožní tisk na mnoha modelech tiskáren, avšak neumožní využi-tí speciálních funkcí těchto modelů.
Foomatic (*http://www.linux-foundation.org/en/OpenPrinting/Database/Foomatic*)Foomatic generuje PPD vhodný pro použití se všemi ovladači tiskáren, které mají veškerépodrobnosti v databázi linux-foundation.org. Lze jej využívat se skriptem foomatic-rip, kterýpoužívají volně šiřitelné ovladače. V současnosti podporuje mnoho tiskáren v tomto systému.Tvoří základ pro podporu nepostscriptových tiskáren ve většině distribucí Linuxu. CUPS a Foo-matic jsou čím dál oblíbenější tiskové systémy a lze je doporučit k širokému použití.

Postscriptové PPD

CUPS může používat soubory PPD dodané prodejcem přímo pro postscriptové tiskárny. Bývа-jí součástí dodávky windowsovských ovladačů tiskáren nebo je lze nalézt na stránkách pro-dejců tiskáren. Máte-li si možnost vybrat si mezi ovladačem pro Windows 9x a Windows NT/W2K, zvolte raději Windows NT. Pro řadu postscriptových tiskáren distribuuje soubory také firma Adobe (*http://www.adobe.com/products/printerdrivers/winppd.html*).

ESP Print Pro

Firma Easy Software Products, Inc. (*http://www.easysw.com/*) prodává CUPS v balíku se sadou proprietárních ovladačů. I když nejsou volně šiřitelné, lze jimi ovládat mnoho běžných tiská-ren. Balík je ve srovnání s cenou jednoho ovladače poměrně drahý, avšak své místo na trhu určitě nalezne. Obsahuje i grafické nástroje.

Gimp-Print

Ovladače Gimp-Print (*http://gimp-print.sourceforge.net/*) jsou vysoce kvalitní ovladače pro tis-kárny Canon, Epson, Lexmark a PCL a je možno je využít se systémy Ghostscript, CUPS, Foo-matic a Gimp.

OMNI (*http://www-124.ibm.com/developerworks/oss/linux/projects/omni/*)

Omni je balík vytvořený IBM, v současné době podporuje více než 450 tiskáren. Ovladač tis-kárny OMNI je distribuován

IBM pod licencí LGPL.HPIJS (*http://hpijs.sourceforge.net/*)HPIJS podporuje v současnosti kolem 150 tiskáren firmy HP ve

vynikající tiskové kvalitě (zatím

pouze prostřednictvím Foomatic). Od verze 1.0.1 byla z licence odstraněna klauzule „hp Pro

duct Only“ a ovladače jsou distribuovány pod licencí BSD.Program XPP (*http://cups.sourceforge.net/xpp/*) dodaný třetí stranou (viz obrázek 11.4) nabízíuživatelské funkcionalitě CUPS velmi zdařilé grafické rozhraní včetně skvělého rozhraní provolby v průběhu tisku (viz obrázek 11.5). Další informace o XPP viz kapitola „XPP“.

LPD

LPD, původní Line Printer Daemon z BSD Unixu, je v unixovém světě standardem již řadu let. Je k dispozici na všech unixových systémech a nabízí minimální množinu funkcí pokrývajících potře-by z doby sdílení počítačů. Navzdory specifické historii je i dnes užitečný jako základní tiskový spooler. K rozumnému použití s moderními tiskárnami vyžaduje specifická nastavení, jako jsou různé filtrační skripty a uživatelská rozhraní. To všechno ale existuje a funguje.
LPD je rovněž název síťového tiskového protokolu, definovaného dokumentem RFC 1179. Tento protokol umí nejen démon LPD, ale v zásadě každý tiskový server, síťové tiskárny i jiné spoolovací systémy. LPD je nejmenší společná množina funkcí standardního síťového tiskového prostředí.

LPRng (viz kapitola „LPRng“) je výrazně lepší implementace standardního LPD návrhu. Pokud musíte používat LPD, zvolte raději LPRng. Vyžaduje o hodně méně magie k tomu, abyste jej donu-tili dělat přesně to, co potřebujete, a všechny čáry jsou dobře dokumentovány. LPRng je v zásadě rozšířenou implementací LPD s vylepšenou bezpečností a doplněnou funkcionalitou.

Po světě se toulá řada různých zdrojových podob LPD. Pravděpodobným oficiálním vlastníkem je nějaký derivát BSD Unixu, nicméně každý vesele implementuje různé změny, které se neznámý-mi způsoby prolínají, a tak lze jen velmi obtížně určit, jakou konkrétní verzi LPD kdo používá. Z těch veřejně dostupných LPD nabízí GNUlpr (*http://sourceforge.net/projects/lpr/*) verzi s několi-ka drobnými úpravami, které vylepšují uživatelské rozhraní. GNUlpr podporuje zadávání para-metrů přepínačem -o, ty se pak předávají filtrům. Je to podobné funkci, nabízené i řadou tradič-ních unixových dodavatelů a analogické (i když nekompatibilní) s mechanismem přepínače -z programu LPRng.

Používáte-li LPD, je nejlepší ovládat jej pomocí vnějšího programu („front-endu“). Je však z čeho vybírat; nejlepší jsou asi KDEPrint, GPR (viz kapitolu „Grafické tiskové nástroje“) a XPP. Existují i další; můžete mi o nich napsat.

LPRng

Někteří prodejci Linuxu dodávají LPRng, což je novější implementace LPD. Mnohem snáze se spra-vuje ve velkých instalacích (rozuměj: více než jedna tiskárna, nějaké sériové tiskárny nebo nějaké neobvyklé síťové non-lpd tiskárny) a není naprogramován tak zmateně jako lpd. Dalo by se o něm dokonce říci, že je bezpečný – neexistují binární soubory s příznakem SUID a podporuje autenti-zaci prostřednictvím PGP a Kerbera.

LPRng také obsahuje některá vzorová nastavení pro běžné síťové tiskárny – zejména laserové tis-kárny HP – které obsahují i účtovací možnosti. LPRng používá víceméně stejný základní model filtrů jako lpd v BSD, takže podpora LPD (*http://www.linuxprinting.org/lpd-doc.html*) na stránkách linuxprinting.org platí i pro LPRng. To umožňuje efektivní využívání volně šiřitelných ovladačů pro více tiskáren.

LPRng je distribuován buď pod licencí GPL nebo Artistic.

PPR

PPR (*http://ppr.trincoll.edu/*) je spooler zaměřený na PostScript, který obsahuje základní post-scriptovou lexikální analýzu, z níž odvozuje řadu užitečných vlastností. Má dobré účtovací mož-nosti, dobrou podporu pro klienty Appletalk, SMB a LPD a mnohem lepší zpracování chyb než lpd. Podobně jako ostatní spoolery může PPR na obsluhu nepostscriptových tiskáren volat Ghost-script.

PPR byl napsán a je používán na Trinity College. Licence je podobná BSD; libovolně použitelný, příspěvek se však považuje za povinnost.

Ostatní PDQ

PDQ znamená „Print, Don't Queue“ (tiskni, nečekej ve frontě) a způsob tisku tomu odpovídá. PDQ je tiskový systém s vestavěnou výkonnou syntaxí pro konfiguraci ovladačů, který nepoužívá tiskového démona. Zahrnuje možnost nastavovat parametry tiskárny a používat grafické nebo řád-kové nástroje, jimiž mohou uživatelé tyto parametry nastavit. Uživateli se zobrazí hezký dialog, ve kterém může nastavit rozlišení, duplexní režim, typ papíru a další.

Spouštění všech filtrů v uživatelském režimu má řadu výhod – prakticky se eliminují bezpečnost-ní problémy vznikající v PostScriptu, efektivně lze tisknout vícesouborové úlohy LaTeXu jako sou-bory dvi a další.

PDQ není bez vad: Ta nejznámější je, že zpracovává celou úlohu před tím, než ji odešle na tis-kárnu. V případě rozsáhlých úloh je PDQ poněkud nepraktický a tisk může skončit kopírování stovek megabajtů tam a zpět na disku. A co je horší, pomalé ovladače, jako např. ovladače kva-litnějších inkoustových tiskáren, ani nezahájí tisk, dokud Ghostscript a ovladač neukončí zpraco-vání. To může trvat i několik minut.

PDQ ovšem má svoje opodstatnění; má jednoduchý návrh, který uživateli nebrání v řízení tisku, nepřekračuje žádné obvyklé bezpečnostní hranice, a není tedy bezpečnostním rizikem, která můžete tak často vidět v jiných systémech. Navíc je malý.

Systém PDQ bohužel nikdo dál nevyvíjí. Pokud by se někdo takový našel, byl by velice vítán.

GNUlpr

GNUlpr spatřil světlo světa jako příspěvek VA Linuxu sponzorovaný firmou HP. Bohužel dnes je už téměř mrtev.

CPS

Coherent Printing System (*http://www.tww.cx/cps.php*) je množina skriptů v Perlu, které se nazý-vají lpr, lpd, lprm a lpq. Nahrazují programy stejného jména, které jsou součástí většiny systé-mů Linux.

CEPS

Cisco Enterprise Print System vyvinul Damian Ivereigh, když pracoval jako správce systému ve firmě Cisco. Udělal víc, než za co byl placen – v zájmu odstranění administrativních nepořádků vyvinul nový tiskový systém. Systém poskytl k volnému použití pod licencí GNU General Public License. Instalace CEPS se však vyplatí jen ve velkých společnostech.

Jak to celé funguje

Abyste si dokázali poradit s různými záludnostmi tisku, je nutné chápat, jak celý tiskový systém funguje. Všechny systémy fungují v zásadě stejně, i když se může poněkud lišit pořadí některých kroků, případně mohou být některé vynechány.

Uživatel odesílá tiskovou úlohu společně s nastavenými parametry. Data úlohy jsou obvykle, byť ne vždycky, PostScript.
Spooler zkopíruje úlohu a parametry přes síť směrem k tiskárně.
Spooler čeká na dostupnost tiskárny.
Spooler uplatní na úlohu uživatelem zvolené parametry a provede překlad dat úlohy do nativního jazyka tiskárny, což typicky není PostScript. Tomuto kroku se říká filtrování a vět-šina problémů s tiskem spočívá právě v dobře nastaveném filtrování.
Úloha je hotova. V této fázi spooler obvykle provede potřebný úklid. Pokud došlo při zpra-cování úlohy k nějaké chybě, spooler typicky uživatele nějakým způsobem (například e-mailem) upozorní.

CUPS

Úlohu pomocí CUPS vytisknete jak pomocí příkazů z BSD (viz kapitolu „LPD“), tak i ze systému
System V, což je pro ty, kdo mají zkušenost alespoň s jedním z těchto systémů, velmi snadné. Původně CUPS postrádal vnitřní program („backend“), jako má LPD, což bylo rychle napraveno. Nyní existují vnitřní programy přinejmenším pro systémy IPP, LPD, SMB, JetDirect, USB, Netatalk k paralelním i sériovým tiskárnám. Můžete si najít i jiné anebo napsat vlastní.

Vestavěných ovladačů, jejichž prostřednictvím byste mohli tisknout na většině tiskáren (avšak nikoli s maximálním rozlišením), existuje jen několik. K systému CUPS můžete přidat soubor PPD pro postscriptový ovladač, chcete-li však tisknout v nejlepší kvalitě na vaší báječné nové tiskárně HP Deskjet, máte smůlu. Snad přinese nápravu Foomatic. Můžete jej zkombinovat s CUPS. Foo-matic ke svým kouzlům používá filtry CUPS, které se jmenují foomatic-rip. K popisu funkcí tiská-ren používají soubory PPD, a to i u nepostscriptových tiskáren. V současnosti je kombinace CUPS
+ Foomatic doporučovaným tiskovým systémem. Některé linuxové distribuce už ji používají

a jejich počet jenom poroste.Plánovač CUPS nejen přijímá úlohy, nýbrž také spravuje internetové rozhraní. Můžete přidávata ubírat tiskárny, rušit úlohy, spouštět a zastavovat tisky. Přenosy tiskových úloh budou k dispo-zici v pozdější verzi.

LPD

LPD znamená Line Printer Daemon a v různých kontextech se touto zkratkou označuje jak samotný démon, tak celá skupina programů, které zajišťují tiskový spooling. Jsou to:

lpd (*http://www.linuxprinting.org/man/lpd.8.html*)

Spoolingový démon. Jedna instance běží a řídí celý tisk na počítači, další instance běží pro

každou tiskárnu v době, kdy tiskne. lpr

(*http://www.linuxprinting.org/man/lpr.1.html*)

Uživatelský příkaz pro tisk. Kontaktuje *lpd* a přidá úlohu do fronty. lpq

(*http://www.linuxprinting.org/man/lpq.1.html*)

Vypíše seznam úloh ve frontě.

lpc (*http://www.linuxprinting.org/man/lpc.8.html*)

Příkaz pro řízení tiskového systému. Umožňuje zastavovat, spouštět, měnit pořadí a podobně

úloh ve frontách. lprm (*http://www.linuxprinting.org/man/lprm.1.html*)

Odstraňuje úlohu z fronty.

Jak to zapadá do sebe? Odehrávají se následující kroky:
Při startu systému se spouští démon lpd. Čeká na spojení a spravuje tiskové fronty.
Uživatel zadává úlohy k tisku příkazem lpr, popřípadě grafickým rozhraním jako GPR, PDQ a podobně; lpr kontaktuje po síti lpd a předá mu datový soubor (obsahující tištěná data) a řídicí soubor (obsahující parametry úlohy).
Když se uvolní tiskárna, spustí hlavní lpd svou další instanci, která obslouží tisk úlohy.
Synovský lpd spustí příslušný filtr (definovaný příznakem if v souboru /etc/printcap) a výsledná data pošle na tiskárnu.

Tento systém byl původně navržen v době, kdy většina tiskáren byly řádkové tiskárny – tedy v době, kdy se převážně tiskly čistě textové soubory. Umístěním všech potřebných kroků do pří-slušných filtrů je možné pomocí lpd vyhovět i nárokům moderního tisku (tedy, alespoň jakžtakž

– řada jiných systémů funguje lépe).

Při tvorbě filtrů pro LPD je možné použít řadu užitečných programů. Jsou to mimo jiné:

gs

Ghostscript je softwarový interpret PostScriptu (označovaný též jako Raster Image Processor nebo RIP). Jako vstup přijímá PostScript a výstup vytváří v různých ovládacích jazycích tiskáren nebo v různých grafických formátech. O tomto programu hovoříme v kapitole „Ghostscript“.

ppdfilt

Samostatná verze jedné z komponent CUPS. Filtruje PostScript, provádí některé základní transfor-mace (jako například řazení více kopií) a doplňuje uživatelem nastavené parametry na základě PPD souboru (PostScript Printer Definition), který bývá obvykle dodáván s tiskárnou.

ppdfilt je nejlépe používat se systémem LPD, který akceptuje volby (podobně jako GNUlpr nebo LPRng) a skriptový filtr, který převede uživatelské volby do ekvivalentu příkazu ppdfilt. Ve VA Linux a v HP existuje upravený balík, který dělá přesně toto; s postscriptovou tiskárnou jsou výsledky velmi dobré. Informace o tomto systému viz kapitola „LPD pro postscriptové tiskárny“.

ps2ps

Jde o skript dodávaný s Ghostscriptem. Filtruje PostScript na jednodušší PostScript na nižší úrov-ni jazyka. Je to užitečné, pokud máte starší postscriptovou tiskárnu, protože většina moderních programů produkuje i moderní PostScript.

mpage

Tento nástroj přijímá vstup v PostScriptu a generuje výstup s více stránkami tištěného textu na jedné tiskové stránce. Tuto funkci nabízí i řada dalších programů, například enscript, nenscript a a2ps.

a2ps

Program a2ps, any-to-ps, přijímá vstup v různých formátech a konvertuje je na PostScript.

Jak všechno nastavit

V případě běžných konfigurací můžete tuto kapitolu pravděpodobně ignorovat a přejít rovnou ke kapitole „Řešení jednotlivých distribucí“ nebo ještě lépe, nahlédnout do dokumentace dodavatele. Většina distribucí Linuxu nabízí jeden či více „blbuvzdorných“ nástrojů, které umožňují za běžných podmínek a při běžných tiskárnách vše dále popsané jednoduše nastavit.

Pokud nástroje dodavatele nefungují nebo pokud byste chtěli interaktivně řídit parametry tisku při tisku, měli byste použít jiný systém. Dobrou volbou je PDQ, nabízí velmi dobrou funkčnost a jed-noduchou konfiguraci. Dalším dobrým systémem je APS Filter, který konfiguruje fronty a filtry LPD na většině unixových systémů.

Dále můžete vyzkoušet rozhraní tiskového systému na stránkách *http://www.linuxprinting.org/*, která umožňují propojit řadu volně šířených ovladačů s různými spoolovacími systémy. Po zavr-šení celého projektu budou tato rozhraní nabízet nejlepší funkčnost – podporují všechny typy volně šířených ovladačů, podporují uživatelská nastavení a všechny běžné spoolery.

Konfigurace CUPS

Používáte-li klienta s CUPS a server CUPS je už zkonfigurovaný, instalace tiskáren na vašem kli entovi už nemůže být jednodušší: Neuděláte vůbec nic. Klient si pomocí všesměrového přenosu najde server CUPS a automaticky zkonfiguruje tiskárny, které jsou nainstalovány na tomto tisko vém serveru. To je funkce, kterou je na velkých sítích nutno skutečně ocenit.

Ruční konfigurace tiskáren s CUPS je také chuťovka. Pokud s takovou prací teprve začínáte, bylo by asi lepší použít internetové rozhraní. Jestliže však je tiskáren, které máte zkonfigurovat, více, bude rychlejší pracovat pomocí příkazových řádků.

URL internetového rozhraní CUPS je implicitně *http.///hostname:631/admin.* Je-li to nutné, v sou-boru cupsd.conf je možno změnit font.Obecná syntaxe přidání tiskárny z příkazového řádku je lpadmin -p printer -E -v device -m ppd.Příkazem lpadmin s volbou -p přidáme nebo změníme tiskárnu. Tiskárny jsou uloženy v soubo-ru. Volbou -x zrušíme uvedenou tiskárnu. Další volby viz manuálové stránky příkazu lpadmin.

Příklad 3. Příklady příkazových řádků

```
/usr/sbin/lpadmin -p testpr1 -E -v socket://192.168.1.9 -m deskjet.ppd /usr/sbin/lpadmin -p testpr2 -E -v parallel:/dev/lp0 -m
laserjet.ppd /usr/sbin/lpadmin -x testpr1
```

Další informace o konfiguraci tiskáren a o volbách naleznete v dokumentaci CUPS (*http://www.cups.org/documentation.html*, přístupná bývá i na *http://localhost:631 v sekci* „Docu-mentation“). Konfigurační postupy jsou také v Software Administrators Manual.

Konfigurace LPD

Hodně linuxových systémů obsahuje LPD. V této kapitole popisujeme základní nastavení LPD, podrobnostem o vytváření složitějších filtrů a o síťovém tisku se věnujeme samostatně.

Základní konfigurace LPD

Základní nastavení LPD vede k systému, který umí řadit úlohy do fronty a tisknout je. Nestará se o to, zda tiskárna bude úlohám rozumět a výsledný vzhled dokumentů asi nebude dokonalý. Někde ale musíme začít. Chcete-li do LPD přidat frontu, musíte upravit soubor /etc/printcap a vytvořit nový adresář pod /var/spool/lpd.

Záznam v /etc/printcap vypadá takto:
```
# LOCAL djet500lp|dj|deskjet:\
        :sd=/var/spool/lpd/dj:\
```

```
:mx#0:\
:lp=/dev/lp0:\
:sh:
```

Tímto definujeme frontu pojmenovanou *lp*, *dj* a *deskjet*, umístěnou v adresáři /var/spool-/lpd/dj, bez limitu velikosti úlohy, ze které se tiskne na zařízení /dev/lp0 a která k úloze nepři-dává hlavičkovou stránku (například s údajem o tom, kdo úlohu vytvořil).
Teď si přečtěte manuálovou stránku printcap (*http://www.linuxprinting.org/man/printcap.5.html*). Výše uvedený příklad vypadá velmi jednoduše, je v něm ale zrada – pokud na tiskárnu DeskJet 500 nebudeme posílat data, kterým bude tiskárna rozumět, budou se tisknout divné věci. Napří-klad pokud vytisknete normální unixový text, bude tiskárna doslovně interpretovat řídicí znaky LF a výsledkem bude:

```
This is line one. This is line two. This is
        line three.
```

a tak dále. Pokud tímto způsobem budeme tisknout PostScript, dostaneme krásný výpis post

scriptových příkazů zobrazený v tomto „schodišťovém“ efektu, výsledek ale k ničemu moc nebude.Zjevně je potřeba něco navíc, a to je právě úkolem filtrace. Pozorný čtenář, který nepřeskočilmanuálovou stránku printcap, jistě postřehl parametry if a of. Nuže, if neboli *input filter* je to,co teď potřebujeme.

Pokud si napíšeme krátký skript, který ke znakům LF doplní i znaky CR, můžeme zrušit schodiš-ťový efekt. Přidáme tedy řádek s parametrem if:lp|dj|deskjet:\

```
:sd=/var/spool/lpd/dj:\
:mx#0:\
:lp=/dev/lp0:\
:if=/var/spool/lpd/dj/filter:\
:sh:
```

Filtrační skript může vypadat například takto:

```
#!perl # The above line should really have the whole path to perl # This script must be executable: chmod 755 filter while(<STDIN>){chomp $_; print "$_\r\n";}; # You might also want to end with a form feed: print "\f";
```

Pokud bychom to všechno udělali, dostaneme frontu, jejímž prostřednictvím můžeme tisknout normální textové soubory a dostaneme použitelné výsledky. (Ano, existují asi čtyři miliony lepších způsobů, jak filtr napsat, ale nejsou tak názorné. Klidně to udělejte efektivněji.)

Posledním zbývajícím problémem je fakt, že tisk prostého textu dneska málokoho nadchne – roz-hodně bychom chtěli tisknout i PostScript, grafiku a podobně. Jistě to jde a je to poměrně jedno-duché. Celá finta spočívá v rozšíření schopností výše uvedeného filtru.
Takovému filtru říkáme „magický“ filtr. Nenamáhejte se s tím psát si jej sami, pokud netisknete vyloženě avantgardní formáty – řada už existuje a mají snadno použitelná interaktivní konfigurační rozhraní. Stačí prostě zvolit vhodný existující filtr:

Foomatic-rip

Foomatic-rip (*http://www.linuxprinting.org/lpd-doc.html*) je filtr používající data z databáze Linux-Printing.org. Podporuje prakticky všechny volně distribuované ovladače tiskáren včetně ghost-scriptových ovladačů, uniprintových ovladačů a různých filtrů, které se potulují po Internetu; foo-matic-rip pracuje s CUPS, LPRng, LPD, GNUlpr, PPR a PDQ bez spooleru.

APS Filter

APS Filter (*http://www.apsfilter.org/*) je filtr určený pro použití v různých Unixech. Podporuje prak-ticky všechny ghostscriptové ovladače. I on pracuje s různými typy LPD, včetně základního BSD i LPRng.

Tiskové filtry RHS

RHS je filtrační systém vytvořený společností Red Hat. Jeho distribuce začala, pokud vím, ve verzi Red Hat 4, kde sloužil jako základ pro grafickou nástavbu printtool pro snadnou konfiguraci tis-káren. Jiné distribuce včetně Debianu dnes distribuují kombinaci rhs-printfilter/printtool. Jedná se asi o nejrozšířenější filtrační systém.

Systém rhs je založen na ASCII databázi, která je jeho součástí. Ta obsahuje údaje o podpoře řady ovladačů Ghostscript a Uniprint, nepodporuje však ovladače filtrového typu. Filtry navíc příliš nepodporují uživatelskou volbu parametrů tisku.

Printtool ukládá v adresáři fronty konfigurační soubor postscript.cfg. V tomto souboru podob-nému shell skriptům představuje každé nastavení jednu proměnnou. V neobvyklých případech můžete užitečné zásahy provést přímo modifikací konfiguračního souboru, kterou nemusí umož-ňovat printtool – typicky půjde o specifikaci neobvyklého ghostscriptového ovladače nebo PPD souboru u VA verze rhs-printfilters.

VA Linux v rámci kontraktu s HP obsahuje některá vylepšení systému rhs-printfilters. Správné verze umožňují volbu parametrů postscriptových tiskáren, které jsou řízeny PPD soubory. O tomto systému hovoříme v kapitole „LPD pro postscriptové tiskárny“.

Použití těchto filtrů má jednu nevýhodu: Starší verze lpd nespouštějí filtr if pro vzdálené tiskár-ny, zatímco většina novějších verzí ano (i když často jen bez parametrů). Verze LPD obsažené v moderních distribucích Linuxu a FreeBSD jej spouštějí, verze LPD ve většině komerčních Unixů ne. Více informací na toto téma uvádíme dále v kapitole věnované tisku v síti. Pokud pracujete jen s lokálně připojenými tiskárnami, tyto problémy se vás netýkají.

LPD pro postscriptové tiskárny

I když většina verzí LPD nezpracovává PostScript dobře (bez ohledu na uživatelská nastavení), VA Linux nedávno modifikoval LPD a filtrační software Red Hat, takže nyní podporuje postscriptové tiskárny poměrně dobře. Vzhledem k tomu, že úmyslem bylo věnovat kód projektu GNU, dostal název GNUlpr (*http://lpr.sourceforge.net/*).

Jak to funguje

Nový systém VA používá soubory PPD – Postscript Printer Definition. PPD soubory poskytuje výrobce tiskárny a obsahují informace o funkcích, které tiskárna umí, společně s postscriptovým kódem, jenž tyto funkce aktivuje. V systému VA funguje klasické schéma LPD poněkud odlišně.
Uživatel může nastavovat parametry příznakem -o. Můžete například zadat -o MediaType: Transparency při tisku na fólii. Je však možné také použít nástavby jako GPR (*http://www.compumetric.com/linux.html*), které umožňují volit parametry v dialogovém rozhraní. Příslušné obrázky můžete vidět v kapitole „Grafické tiskové nástroje“.
LPR předá parametry LPD jako rozšířené atributy v řídicím souboru LPD.
Modifikovaná verze rhs-printfilters obdrží rozšířené parametry v proměnných prostředí a pomocí ppdfilt je přidá k tiskovým datům.

Získání a instalace

RPM balíky nebo zdrojové kódy můžete získat na domovské stránce projektu na SourceForge (*http://sourceforge.net/projects/lpr*). Informace o instalaci najdete v instalačním dokumentu micro-HOWTO (*http://printing.sourceforge.net/gpr-libppd-uhowto.html*). V zásadě musíte odinstalovat původní Red Hat verzi balíků printtool, lpd a rhs-printfilters a nainstalovat jejich VA verze spo-lečně s balíky ppdfilt, gpr a s dalšími nástroji.

Dále potřebujete PPD soubor pro svou postscriptovou tiskárnu. Tyto soubory se obvykle dají snadno najít. VA Linux a HP distribuují PPD soubory pro řadu modelů Laserjet. Další výrobci poskytují PPD soubory ke svým tiskárnám, Adobe distribuuje PPD soubory pro řadu tiskáren (*http://www.adobe.com/products/printerdrivers/winppd.html*).

V současné době je instalace většiny nástrojů poměrně obtížná. Budoucí instalační nástroje mají být postaveny nad konfigurační knihovnou tiskáren, libprinterconf, která umožňuje automatickou detekci a konfiguraci rhs-printfilter pro síťové i lokální tiskárny.

> Je možné použít GPR i samostatně, bez modifikovaného LPD, a dokonce i bez rhs-print-filters. GPR je možné přeložit s podporou kompletního zpracování postscriptových úloh. Tato možnost může být z hlediska instalace jednodušší pro uživatele, kteří nepotřebují tisk-nout přímo přes lpr.

Nastavení parametrů PostScriptu

Po nastavení LPD systému s podporou PostScriptu můžete parametry tiskáren nastavovat dvěma způsoby:

Pomocí grafického rozhraní

> Abyste mohli použít GPR, musíte nejprve vybrat správný soubor PPD. Pak budou na záložce Advanced k dispozici parametry tiskárny. Základní parametry ppdfilt najdete na záložce Common.

Z příkazového řádku Tato verze lpr podporuje přepínač -o. S jeho pomocí můžete zadat jakoukoliv dvojici para-metr/hodnota podle PPD souboru vaší tiskárny. Předpokládejme například, že soubor PPD obsahuje následující:

```
*OpenUI *PrintQuality/Print Quality: PickOne
*DefaultPrintQuality: None
*OrderDependency: 150 AnySetup *PrintQuality
*PrintQuality None/Printer Setting: ""
*PrintQuality Quick/QuickPrint: "<< /DeviceRenderingInfo ...
*PrintQuality Normal/Normal: "<< /DeviceRenderingInfo << /...
*PrintQuality Pres/Presentation: "<< /DeviceRenderingInfo ...
*PrintQuality Image/1200 Image Quality: "<< /DeviceRenderi...
*CloseUI: *PrintQuality
```

Parametr PrintQuality povoluje hodnoty Quick, Normal, Pres a Image. Na příkazovém řádku můžete zadat:

```
% lpr -o PrintQuality:Image file.ps
```

Kromě toho existuje řada voleb společných pro všechny tiskárny, ty budou fungovat stejně jako volby definované v souboru PPD. Jsou to například:

page-ranges

Rozsah stránek pro tisk, například page-ranges:2-3 page-set

Umožňuje tisknout pouze liché (odd) nebo sudé (even) stránky. Například pageset: odd number-up

Na každý list můžete vytisknout více stránek, například number-up:2

Další parametry najdete na manuálové stránce ppdfilt.

Přístupová práva k souborům

Na základě spousty žádostí uvádím výpis přístupových práv u důležitých souborů na svém systé-mu. Nastavení lze provést i lepšími způsoby, například pomocí příznaku SGID pro binární pro-gramy, nemusí být vše nastaveno SUID root. Jde o implicitní nastavení systému a funguje dobře. (Upřímně řečeno, pokud by implicitní nastavení nefungovalo, je čas na změnu dodavatele...)

-r-sr-sr-x 1 root lp /usr/bin/lpr* -r-sr-sr-x 1 root lp /usr/bin/lprm* -rwxr--r--1 root root /usr/sbin/lpd* -r-xr-sr-x 1 root lp /usr/sbin/lpc* drwxrwxr-x 4 root lp /var/spool/lpd/ drwxr-xr-x 2 root lp /var/spool/lpd/lp/

Lpd musí běžet jako root, aby se mohl připojit na nízký síťový port, určený pro službu lpd. Po připojení na port by měl přejít na UID lp.lp nebo tak nějak, ale nemyslím, že to dělá. I to je jeden z důvodů, proč nepoužívat standardní BSD LPD.

Velké instalace

Velké instalace, čímž myslím sítě s více než dvěma tiskárnami nebo počítači, mají speciální poža

davky. Dále uvádíme několik doporučení.

Pro velké sítě je velice vhodný systém CUPS, neboť má řadu velmi dobrých vlastností. Namátkou jmenujme alespoň třídy tiskáren, řízení přístupu a automatickou konfiguraci klienta.

V opravdu velkých prostředích je problémem už samotná distribuce nastavení printcap a filtrů.Tyto problémy řeší Cisco Enterprise Print System (*http://ceps.sourceforge.net/*) a představuje takdobrý začátek, případně téměř kompletní řešení, v závislosti na skutečných potřebách. Střednía velká prostředí si mohou vystačit s nativními funkcemi LPRng nebo CUPS.

Každá tiskárna by měla mít jedno řídicí místo, odkud může administrátor pozastavovat, řadit a přesměrovávat frontu. Lze to implementovat tak, že vše budete tisknout na lokální server, který pak úlohy řadí a směruje je na správné tiskárny. Rozsáhlejší prostředí nebo distribuované sítě mohou používat samostatný server pro každou budovu nebo jinou jednotku.

Používejte CUPS nebo LPRng, přinejmenším na serverech – BSD LPD má příliš mnoho chyb na „opravdové“ použití. Nemusíte mi ale věřit – vyzkoušejte si různé spoolery a zjistíte, který vám vyhovuje nejvíce.

Klientské systémy by neměly používat vlastní tiskové konfigurace. V systému CUPS je možno v jedné podsíti provést klientskou konfiguraci tiskáren automaticky. CUPS dokon-ce můžete zkonfigurovat (BrowsePoll) tak, aby vybral příslušné tiskárny v jiných podsítích, čímž ušetříte klientovi práci s konfigurací. Tuto funkci lze implementovat pomocí rozšíře-né syntaxe printcap v LPRng, takže můžete mít všude stejný printcap. CUPS to řeší pomocí distribuované databáze.

Tiskové fronty by se neměly jmenovat podle typu tiskárny, ale nějak rozumně, například podle umístění tiskárny nebo podle možností tiskárny. Za tři roky, až se tiskárna pokazí, ji budete moci nahradit jinou a nedojde ke zmatkům.

Vytvořte webové stránky s podrobnými informacemi o tiskárnách, včetně jejich umístění, možností a podobně. Mohly by případně vypisovat i obsah tiskové fronty a umožnit sma-zání úloh z fronty. Ve složitých síťových prostředích není možné pracovat bez dostatečné dokumentace.

Používejte internetovou stránku, která zobrazuje podrobné informace o všech tiskárnách včetně umístění, funkcí apod. Měla by se zobrazovat i fronta s tlačítkem pro odstranění úlohy z této fronty. Bez odpovídající dokumentace není možné nastavit komplexní síťové prostředí.

Na systémech Windows a Apple používejte všude buď konkrétní ovladače konkrétních tis-káren (Samba podporuje mechanismus automatické aktualizace ovladačů) anebo ještě lépe, používejte všude obecný postscriptový ovladač. Nikdy nemíchejte různá nastavení, jednoduché textové editory často vytvářejí na různých ovladačích různé výstupy a uživate-lé se těžko vyrovnávají s tím, když se výsledná podoba výtisku mění pro každou dvojici klient/tiskárna.

Pokud je to možné, tisknete-li velké objemy dokumentů, kupte si velkokapacitní tiskárnu. Pokud na to nemáte, využijte funkci více tiskáren na jedné frontě, podporovanou LPRng, a obstarejte si chůvu – tiskárny jsou složitá mechanická zařízení a v intenzivním provozu mají tendenci papír zmuchlat nebo spotřebovat...

Nevěřte tomu, že tiskárna musí být připojena k počítači, ethernetové „tiskové servery“ stojí méně než 100 dolarů. Možnost umístit tiskárnu kdekoliv, kde je síť, je velká výhoda oproti vynucenému umístění poblíž počítače. Snadno pak tiskárny soustředíte na jedno centrální místo.

Používejte SNMP nebo jiné monitorovací mechanismy, které jsou k dispozici – snadno pak bude moci pověřená osoba doplňovat papír, toner a podobně. Správu tiskáren s podpo-rou SNMP umožňuje například npadmin (viz kapitolu „npadmin“).

Účtování

Standardní LPD nabízí jen malou pomoc při účtování tisku. V parametru af souboru printcap můžete definovat jméno účtovacího souboru, to se však pouze předává jako parametr filtru if. Sami pak musíte zajistit, aby filtr provedl zápis do účtovacího souboru, a sami jej pak musíte zpra-covávat (tradiční formát je vhodný zejména pro řádkové tiskárny a špatně se analyzuje v Perlu, takže není důvod se jej držet). Pokud jako filtr použijete program foomatic-rip, musíte jej také upravit, protože vyžaduje specifikaci účtovacího souboru v konfiguračním souboru.
CUPS účtuje stránky tak, že úlohy předává prostřednictvím filtru pstops, který předpokládá vstup v PostScriptu.
„Normální“ (nepostscriptové) úlohy započítává jako 1 stránku, což znamená, že při tisku z windowsového klienta s normálním ovladačem nebude účtování fungovat.

Ghostscript nabízí operátor PageCount, který lze použít k určení počtu stránek v každé úloze
– typicky pouze na konec každé postscriptové úlohy doplníte pár řádků pro zápis údajů do účto
vacího souboru. Pěkný příklad najdete v souboru unix-lpr.sh v distribuci Ghostscriptu. Implementace účtování unix-lpr provádí zápis do účtovacího souboru přímo z interpretu Ghost-scriptu a není tak použitelná s doporučeným parametrem -dSAFER. Lepší řešení by bylo po skon-čení úlohy zjistit počet stránek PJL dotazem na tiskárnu nebo napsat v PostScriptu kód, který vypí-še počet stránek na stdout – zde jej lze snadno odchytit a zpracovat bez nutnosti zápisu do sou-boru.

Spooler LPRng obsahuje příklad implementace účtování pro tiskárny HP – předpokládám, že používá PJL dotazy na tiskárny. Tento způsob by měl fungovat u většiny PJL, postscriptových a SNMP tiskáren, které podporují obousměrnou komunikaci.

Pokud máte síťovou tiskárnu s podporou SNMP, můžete pomocí programu npadmin zjistit počet stránek po skončení každé úlohy. Tato metoda by měla spolehlivě fungovat pro všechny typy úloh. Další informace o programu npadmin najdete v kapitole „npadmin“.

Řešení jednotlivých distribucí

Tato kapitola je ze své podstaty neúplná. Klidně mi pošlete informace o své oblíbené distribuci. Kromě toho existuje řada samostatných balíků, které usnadňují konfiguraci tiskáren v Unixu. Hovoříme o nich v kapitole „Jak všechno nastavit“, najděte si část odpovídající vámi používanému tiskovému spooleru.

Red Hat

Red Hat obsahuje grafický nástroj pro administraci tiskáren, *printtool*, který umožňuje přidávatvzdálené i síťové tiskárny. Umožňuje zvolit typ Ghostscriptem podporovaných tiskáren a dále zaří-zení, na něž se bude tisknout. Pak v /etc/printcap definuje tiskovou frontu a používá filtračníprogram z balíku rhs-printfilters k podpoře PostScriptu a dalších běžných vstupních typů. Totořešení funguje poměrně dobře a v běžných situacích se velmi snadno nastavuje.
Red Hat 6.x obsahuje BSD verzi LPD, Red Hat 7.x používá standardně LPRng. S Red Hatem 6.x a 7.x narazíte v okamžiku, kdy máte tiskárnu, kterou jejich standardní Ghostscript nepodporuje (používají GNU Ghostscript namísto Aladdin Ghostscriptu, takže je podporo-váno méně tiskáren). Podívejte se do seznamu podporovaných tiskáren (*http://www.linuxprinting.org/printer_list.cgi*), kde zjistíte, zda standardní Red Hat vaši tiskárnu podporuje.
Pokud ne, můžete si nainstalovat Aladdin Ghostscript a balíky lpdomatic nebo apsfilter, které vědívšechno o tiskárnách podporovaných nejnovějším Ghostscriptem i o některých dalších.V Red Hat Linuxu 8.0 je stále implicitně instalován LPRng, i když si můžete vybrat CUPS. Avšak
i když explicitně vyberete pouze CUPS, zůstává LPRng nainstalovaný. Ve verzi 8.1 by konečně měl
být CUPS implicitním spoolerem.
Ve verzi Red Hat 9.0 je CUPS konečně implicitní.

Debian

Debian nabízí volbu mezi standardním LPD, LPRng a CUPSem. LPRng a CUPS jsou zjevně lepší volby. PDQ najdete v nestabilní verzi. Dále Debian nabízí různé konfigurační nástroje, nejlepší volbou je asi *APS Filter* verze 5 nebo vyšší, protože obsahuje podporu pro ovladače uniprint LPRng a Ghostscriptu. Podporován je i printtools od Red Hatu pro ty uživatele, kteří mají rádi grafické nástroje.

SuSE/openSUSE

Tiskový systém v SuSE Linuxu je založen na *APS Filter* s některými vylepšeními. *APS Filter* od SuSE rozeznává všechny běžné formáty souborů (včetně HTML, pokud je nainstalován program html2ps). V systémech SuSE jsou dvě možnosti nastavení tiskáren:

YaST umožňuje nastavit tiskárny „PostScript“, „DeskJet“ a „Other“, podporované ovladači Ghostscript, kromě toho je možné nastavit tiskárny HP GDI (DeskJet 710/720, 820, 1000, pomocí balíku ppa). YaST zapíše do /etc/printcap údaje pro každou tiskárnu („raw“, „ascii“, „auto“ a „color“, pokud jde o barevnou tiskárnu). Dále YaST vytvoří adresář fronty a nastaví soubory

apsfilterrc, kde můžete doladit některá nastavení (preload Ghost scriptu, velikost a orientace papíru, rozlišení, escape sekvence tiskárny a podobně). Pomo-cí YaST je možné nastavovat i síťové tiskárny (TCP/IP, Samba a Novell NetWare).
Kromě toho SuSE obsahuje i standardní program SETUP z původního balíku *APS Filter* (s některými vylepšeními). Tento konfigurační skript spustíte příkazem lprsetup. Jakmile si zvyknete na jeho rozhraní, budete moci konfigurovat lokální i síťové tiskárny.

Instalační manuál SuSE popisuje oba postupy instalace.

Wolf Rogner ohlásil se SuSE některé problémy. Vadit mohou následující věci:

Klasický skript SETUP z *APS Filter* není v pořádku stejně jako konfigurační nástroj v KDE. Použijte YaST. [Ed: Už to funguje? Od ohlášení už uplynulo dost času.]
U síťových tiskáren ovládaných přes Ghostscript budete muset nejprve odkomentovat řádek REMOTE_PRINTER=“remote“ v /etc/apsfilterrc. Pak pomocí YaST nastavte tis-kárnu a pomocí konfigurace sítě nastavte vzdálenou tiskovou frontu.
YaST nepodporuje barevné laserové tiskárny, takže nastavte černobílou tiskárnu a pak všude v printcap změňte mono na color. Možná budete muset přejmenovat i adresář fronty.

Mandrake/Mandriva

Ve verzi 7.2 používá Mandrake standardně CUPS. Administrativní rozhraní poskytuje program QtCUPS. Snažili se poskytnout co možná nejvíce ovladačů a používají PPD soubory pro CUPS, generované mým vlastním rozhraním foomatic (*http://www.linuxprinting.org/foomatic.html*).

Starší verze Mandrake, pokud vím, používaly stejné nástroje jako Red Hat.

Slackware

Slackware obsahuje *APS Filter*. Skript SETUP je nainstalován jako program apsfilterconfig. Rozumné nastavení byste měli být schopni vytvořit prostým spuštěním tohoto programu. Slackware 9.0 obsahuje i CUPS, ale implicitní je stále LPRng + APS Filter.

Ghostscript

Ghostscript (*http://www.cs.wisc.edu/~ghost/*) je nesmírně významný program pro softwarově říze-ný tisk. Většina programů v Unixu generuje výstup v PostScriptu, jehož podpora je u většiny tis-káren (draze) placený doplněk. Ghostscript je naproti tomu zadarmo a z postscriptového vstupu vygeneruje výstup v jazyce, kterému rozumí vaše tiskárna.

Ghostscript existuje v několika podobách. Komerční verzi Ghostscriptu, Aladdin, lze zadarmo používat pro osobní potřebu, nesmí však být distribuována komerčními entitami. Tato verze je zhruba rok napřed před free verzí. Momentálně například podporuje řadu barevných inkousto-vých tiskáren, které starší verze neznají, a má také výrazně lepší podporu PDF.

Hlavní free verzí Ghostscriptu je GNU Ghostscript, což je jednoduše stará verze Aladdin Ghost scriptu. Díky tomuto poněkud podivnému uspořádání je Aladdin samofinancující se free projekt. Vývoj vede L. Peter s několika zaměstnanci a nové verze licencuje dodavatelům hardwaru a soft-waru k použití v komerčních produktech. I když toto uspořádání umožňuje L. Peterovi už řadu let pracovat na Ghostscriptu, bohužel znemožňuje spolupráci s širší vývojářskou komunitou. Toto řešení zejména nevyhovuje autorům ovladačů. Plány L. Petera na odchod do důchodu předpo-kládají zapojení širší komunity, takže zvažuje změnu licence a vytvořil projekt na SourceForge.

Třetí verzí Ghostscriptu je ESP Ghostcript, udržovaný společností Easy Software Products (autoři CUPS) na základě kontraktu se společností Epson. ESP Ghostscript je kombinací ovladačů gimp print projektu a GNU Ghostscriptu plus vybraných doplňků. Tato verze ještě není plně funkční, ale brzy by měla být vylepšena a měla by usnadnit život majitelům tiskáren s ovladači gimp-print.

Ať už gs (*http://www.linuxprinting.org/man/gs.1.html*) používáte jakkoliv, rozhodně jej vždy spou-štějte se zákazem přístupu k souborům (parametr -dSAFER). PostScript je plnohodnotný progra-movací jazyk a škaredý program v PostScriptu vám může způsobit řadu problémů.

Abychom nezapomněli, PDF, Portable Document Format od Adobe, je (přinejmenším od verze 1.3) pouze o trochu více než komprimovaný PostScript. Ghostscript umí zpracovávat PDF stejně jako PostScript. Možná tak budete první v okolí, kdo umí přímo tisknout PDF.

Spuštění Ghostscriptu

Typicky se Ghostscript spouští filtrem, který používáte (doporučuji Foomatic), pro účely ladění je však často užitečné spouštět jej přímo; gs -help vypíše stručný seznam parametrů a dostupných ovladačů (jsou uvedeny ovladače, s nimiž je Ghostscript přeložen, nejde o úplný seznam všech podporovaných ovladačů).

Pro účely testování budete gs spouštět například takto:

```
gs <parametry> -q -dSAFER -sOutputFile=/ dev/lp1 test.ps
```

Doladění výstupu Ghostscriptu

Existuje řada věcí, které můžete podniknout, pokud není výstup Ghostscriptu uspokojivý (může te vlastně provést naprosto *cokoliv*, protože máte k dispozici zdrojový kód). Některé z těchto možností jsou popsány v Ghostscript User Guide (soubor Use.htm,*http://www.cs.wisc.edu/~ghost/aladdin/doc/Use.htm*, v distribuci Ghostscriptu, měli byste jej najítv /usr/doc nebo /usr/share/doc). Vybrané možnosti můžete použít jako parametry ovladačeve filtračním systému.

Velikost a umístění výstupu

Umístění, velikost a poměr stran obrázku na stránce se v Ghostscriptu řídí ovladači specifickými pro danou tiskárnu. Pokud zjistíte, že tištěné stránky jsou příliš krátké, příliš dlouhé nebo dvojná-sobně veliké, podívejte se do zdrojového kódu ovladače a upravte parametry, které uznáte za vhodné. Bohužel, každý ovladač je jiný, takže neumím říct, co přesně kde nastavit, nicméně vět-šina ovladačů je rozumně dokumentovaná.

Gamma, velikost bodu atd.

Většina tiskáren jiných než laserových trpí tím, že mají velké tiskové body. Výsledkem jsou příliš tmavé obrázky. Pokud máte tento problém s jinak neupravitelným ovladačem, můžete zkusit pou-žít vlastní transformační funkci. Vytvořte následující soubor v adresáři lib Ghostscriptu a jeho název přidejte při spouštění gs před název tisknutého souboru. Možná budete muset upravit kon-krétní hodnoty tak, aby vyhovovaly vaší tiskárně. Zmenšíte-li je, tisk bude světlejší. Zejména pokud ovladač používá k rastrování barev Floyd-Steinbergův algoritmus, bude rozumné použít menší hodnoty (0,2–0,15).

```
%!
%transformační funkce pro CMYK

{0.3 exp} {0.3 exp} {0.3 exp} {0.3 exp} setcolortransfer
```

Nastavením těchto hodnot můžete „opravit“ tiskárnu, které netiskne určitá barva. Pokud se bude-te o něco takového snažit, doporučuji jako test soubor colorcir.ps, dodávaný s Ghostscriptem (v adresáři examples).

U řady novějších barevných ovladačů existují řádkové parametry nebo konfigurační soubory, které umožňují pomocí gamma korekce a dalších nastavení uzpůsobit tiskárnu různým typům papíru. Než začnete nevyhovující tisky opravovat přímo zásahy do PostScriptu, zkuste nejprve tato nastavení.

Barevný tisk v Ghostscriptu

Barevný dithering je v Ghostscriptu implicitně optimalizován pro zařízení s nízkým rozlišením. Dit-hering je poměrně bídný díky snaze vytvářet výstup 60 PPI (ne DPI, ale PPI – počet „zdánlivých“ barevných bodů na palec po provedení ditheringu). Na moderních barevných tiskárnách je výstup nehezký, zejména inkoustové tiskárny s kvalitním papírem jsou schopny tisknout s mnohem lep-ším PPI.

Můžete to upravit parametrem -dDITHERPPI=x, kde x je požadovaná hodnota. Nemusí to fun-govat pro všechny ovladače, řada novějších ovladačů (například ovladač stp pro Epson Stylus) používá vlastní dithering a tento parametr nebere na vědomí. Jiné ovladače mohou používat buď standardní ghostscriptový dithering nebo svůj vlastní (například bjc600 pro tiskárny Canon Bubb-lejet).

Dithering v Ghostscriptu je poměrně hloupý. Jádro Ghostscriptu jednoduše nepodporuje řadu věcí potřebných pro kvalitní tisk na moderních tiskárnách. Řada projektů, které by tuto situaci řešily (a svět otevřeného softwaru je schopen tato řešení poskytnout), je brzděna licenční politikou Ghostscriptu a jeho uzavřeným vývojem. Na setkání *Open Source Printing Summit 2000* (*http://www.linuxprinting.org/summit.html*) tuto situaci řešili všichni, jichž se to týká, a dá se čekat, že v brzké době dojde ke zlepšení.

Sítě

Jednou z funkcí většiny spoolerů je, že podporují přes síť tisk na tiskárny fyzicky připojené k jiným počítačům nebo přímo do sítě. Při vhodné kombinaci filtračních skriptů a vybraných nástrojů můžete transparentně tisknout na tiskárny ve všech typech sítí.

Tisk na unixové lpd stroje

Aby mohly vzdálené počítače tisknout na vaší tiskárně protokolem LPD, musíte je uvést v soubo-ru /etc/hosts.equiv nebo /etc/hosts.lpd. (Pozor na to, že soubor host.equiv způsobuje celou řadu dalších efektů; než sem nějaký počítač uvedete, měli byste dobře vědět, co děláte.) Pomocí příznaku rs můžete povolit tisk jenom určitým uživatelům vzdáleného systému – přečtě-te si manuálové stránky lpd.

Když používáte lpd

Abyste mohli tisknout na vzdáleném počítači, záznam v /etc/printcap by měl vypadat nějak takto:

```
# REMOTE djet500
lp|dj|deskjet:\
        :sd=/var/spool/lpd/dj:\
        :rm=machine.out.there.com:\
        :rp=printername:\
        :sh:
```

V tomto uspořádání stále existuje i lokální fronta spravovaná démonem lpd. Pokud by vzdálený stroj nebyl dostupný, budou úlohy čekat v lokální frontě, dokud je nebude možné odeslat.

Když používáte rlpr

Pomocí rlpr můžete odeslat tiskovou úlohu přímo do vzdálené fronty, aniž byste museli konfigu-rovat lpd. Je to užitečné zejména v situacích, kdy pouze čas od času tisknete na různé tiskárny. Z dokumentace k rlpr uvádíme:

„Rlpr používá TCP/IP k odeslání tiskových úloh na lpd servery kdekoliv v síti.“ Na rozdíl od lpr rplr *nevyžaduje*, aby lokální počítač „znal“ vzdálenou tiskárnu, na níž chcete tisk-nout (například prostřednictvím /etc/printcap), a je tedy považován za pružnější řešení s men-šími nároky na administraci. Je možné použít rlpr kdekoliv, kde se používá klasické lpr,a je s ním zpětně kompatibilní. Hlavní výhodou rlpr je možnost vzdáleně tisknout *odkudkoliv kdeko-liv* bez ohledu na to, jak je nakonfigurován systém, z nějž tisknete. Může fungovat jako filtr stej-ně jako tradiční lpr, takže vzdálení klienti jako netscape, xemacs atd. mohou velmi jednoduše tisk-nout na lokální tiskárně.
Balíček s rlpr je k dispozici na Metalabu, ftp://metalab.unc.edu/pub/Linux/system/printing/.

Tisk na Windows nebo Sambu

V praktickém návodu „Sdílení tisku mezi systémy Windows a Debian“ z této knihy najdete více informací než zde.

Když používáte lpd

Je možné směrovat tiskovou frontu přes program smbclient (součást balíku samba, *http://www.linuxprinting.org/man/smbclient.1.html*) na tiskovou službu SMB na protokolu TPC/IP. Samba obsahuje skript smbprint, který to zajišťuje. Stručně řečeno, v adresáři fronty pří-slušné tiskárny vytvoříte konfigurační soubor a jako filtr if nastavíte skript smbprint.

Záznam v /etc/printcap bude vypadat takto: lp|remote-smbprinter:\

```
:sh:\
:lp=/dev/null:\
:sd=/var/spool/lpd/lp:\
:if=/usr/local/sbin/smbprint:
```

Přečtěte si dokumentaci uvedenou ve skriptu smbprint, kde naleznete více informací o potřeb-ných nastaveních.Kromě toho můžete použít smbclient a směrovat soubor přímo na tiskovou službu SMB, bez pou-žití lpd. Viz manuálové stránky.

Tisk na NetWare

Balík ncpfs obsahuje program nprint, který poskytuje stejné funkce jako smbprint, ovšem v sítích NetWare. Balík ncpfs můžete získat na Metalabu, ftp://metalab.unc.edu/pub/Linux/sys-tem/filesystems/ncpfs/. Z dokumentace pro verzi 0.16 uvádíme:

S pomocí ncpfw můžete v Linuxu připojovat svazku na serveru NetWare. Můžete také tisknout do tiskových front NetWare nebo fronty NetWare obsluhovat spoolerem v Unixu. Potřebujete jádro
1.2.x nebo 1.3.54 nebo vyšší; ncpfs nefunguje s jádry 1.3.x nižšími než 1.3.54.

Když používáte lpd

Abyste mohli nprint použít s lpd, napište si malý skript, který bude tisknout stdin na tiskárnu NetWare, a ten nainstalujte jako filtr if pro frontu lpd. Dostanete něco jako: sub2|remote-NWprinter:\

```
:sh:\
:lp=/dev/null:\
:sd=/var/spool/lpd/sub2:\
:if=/var/spool/lpd/nprint-script:
```

Skript nprint by mohl vypadat přibližně takto:#! /bin/sh# Nejprve zkuste účet guest bez hesla!/usr/local/bin/nprint -S net -U name -P passwd -q printq-name -

Tisk na tiskárny EtherTalk (Apple)

Balík netatalk obsahuje něco podobného jako nprint a smbclient. Postup tisku na a ze sítí Apple už byl popsán daleko lépe, než bych to dokázal já. Podívejte se na dokument Linux Netatalk HOWTO (*http://thehamptons.com/anders/netatalk/*).

Tisk na síovou tiskárnu

Řada tiskáren obsahuje přímo ethernetové rozhraní a je možné na ně přímo tisknout, typicky pro-tokolem LPD. Držte se instrukcí, které jsou přiloženy u tiskárny nebo jejího síťového adaptéru, nic-méně obecně platí, že na těchto tiskárnách „běží“ lpd a nabízí jednu nebo více front, na které můžete tisknout. Tiskárny HP mohou fungovat například s takovouto konfigurací:

```
lj-5|remote-hplj:\ :sh:\
        :sd=/var/spool/lpd/lj-5:\
        :rm=printer.name.com:\
        :rp=raw:
```

Tiskárny HP LaserJet s rozhraním JetDirect obecně nabízejí dvě tiskové fronty – raw, která umí tisknout PCL (a případně PostScript), a text, jenž zpracovává čisté ASCII (a umí se automaticky vyrovnat se „schodišťovým“ efektem). Pokud máte tříportový JetDirect Plus3, fronty se jmenují raw1, text2 a tak dále.

Společnost ISS identifikovala nebezpečí DoS útoku, kterým je možno zablokovat rozhraní tiskáren HP JetDirect. Většina z nich byla popsána na podzim 98. Tento typ problémů je u podobných zaří-zení poměrně běžný, obecně by neměla být přístupná z běžného Internetu.

Ve velkých prostředích, zejména tam, kde některé tiskárny nepodporují přímo PostScript, je rozumné zřídit vyhrazený tiskový server, na nějž budou všechny počítače tisknout a na kterém poběží všechny ghostscriptové úlohy. Díky tomu bude možné s frontami manipulovat příkazy topq a lprm.

Navíc bude takovýto server udržovat fronty jednotlivých tiskáren, takže uživatelé „vytisknou“ úlohy okamžitě a budou moci pokračovat v práci, aniž by museli čekat, než skončí tisk jiných úloh jiných uživatelů. Doporučuje se to také, pokud máte starší JetDirect, snižuje se zaseknutí tiskárny.

Provedete to tak, že na tiskovém serveru vytvoříte frontu, která bude tisknout přímo na HP LJ se síťovým rozhraním, a všechny klienty v síti nastavíte tak, aby tiskli na LPD frontu na tomto serveru. Některé síťové tiskárny HP zjevně ignorují nastavení úvodní stránky, které posílá klient. Tisk inter-ně generované úvodní stránky můžete vypnout tak, že se na tiskárnu připojíte telnetem, dvakrát zmáčknete Enter, napíšete „banner: 0“ a „quit“. Tímto způsobem můžete měnit i jiná nastavení, jejich seznam získáte příkazem „?“.

Všechna nastavení je možné měnit programem *webJetAdmin* od HP (*http://www.hp.com/ go/ webjetadmin*). Tento balík běží jako démon a na zvoleném portu reaguje na HTTP požadavky. Nabízí formuláře a applety, kterými je možné řídit HP tiskárny v síti. Teoreticky může ovládat i uni-xové tiskové fronty, provádí to ale přes službu *rexec*, která není nijak zabezpečena. Použití této funkce vám rozhodně nedoporučuji.

Tisk na zařízení AppSocket

Některé tiskárny (a některé „tiskové servery“) podporují pouze ubohý pseudoprotokol, komuni-kující přímými TCP spojeními, někdy se označuje jako protokol AppSocket. Do této kategorie spa-dají zejména starší modely karet JetDirect (a některé JetDirectEx). V zásadě se tiskne tak, že musí-te navázat TCP spojení s tiskárnou na určitém portu (typicky 9100 nebo 9100, 9101 a 9102 u tří-portových karet) a po tomto spojení odeslat tiskovou úlohu. LPRng obsahuje podporu odeslání úlohy na libovolný port, u BSD LPD to není tak jednoduché. Asi nejlepším řešením je použít nástroj netcat.

Pokud by to nefungovalo, je možné tisk implementovat mimo jiné i pomocí Perlu programem, který uvádíme dále. Lepší výkon dosáhnete s programem netcat (nc), který dělá téměř to samé obecnějším postupem. Ve většině distribucí by měl být tento program k dispozici.

Spuštění if pro vzdálené tiskárny ve starém LPD

Zvláštností starších verzí lpd je to, že pro vzdálené tiskárny nespouští rozhraní if. (Verze od 0.43 nebo tak nějak jsou upraveny na základě změn ve FreeBSD a if už se spouští vždy.) Pokud nara-zíte na to, že potřebujete pro vzdálenou tiskárnu spustit if a vaše verze lpr to neumí, můžete to vyřešit tak, že vytvoříte dvě fronty a budete úlohy předávat mezi nimi. Podívejme se na následu-jící konfiguraci v printcap:lj-5:\

```
        :lp=/dev/null:sh:\
        :sd=/var/spool/lpd/lj-5:\
        :if=/usr/lib/lpd/filter-lj-5: lj-5-remote:sh:rm=printer.name.com:\
        :rp=raw:sd=/var/spool/lpd/lj-5-raw:
```

ve světle skriptu filter-lj-5:

```
#!/bin/sh gs <options> -q -dSAFER -sOutputFile=- - | \ lpr -Plj-5-remote -U$5
```

Volba -U u lpr funguje pouze tehdy, je-li lpr spuštěn jako démon, a slouží k nastavení správné-ho jména vlastníka úlohy v sekundární frontě. Možná byste měli použít nějakou spolehlivější metodu získání uživatelského jména, protože ne ve všech případech je předáváno jako pátý para-metr. Viz manuálové stránky printcap (*http://www.linuxprinting.org/man/printcap.5.html*).

Tisk z Windows

Tisk z klientů Windows (a případně OS/2) je podporován přímo protokolem SMB z balíku Samba,
který podporuje rovněž sdílení souborových systémů Unixu ve Windows. Samba obsahuje vyčerpávající dokumentaci a rovněž dokument Samba FAQ, který pokrývá nej-častější způsoby použití. Můžete buď nakonfigurovat filtr na Unixu a tisknout na něj PostScript nebo můžete na všech stanicích Windows nainstalovat ovladače příslušné tiskárny a tisknout přímo bez filtru. Použití nativních ovladačů Windows může v některých případech produkovat lepší výsledky, je to ale administrativně náročnější, zejména pokud je stanic mnoho. Zkuste tedy nejprve PostScript. Nové verze Samby podporují mechanismus automatického stažení ovladače, poskytovaný servery Windows NT.

Nějaké informace najdete také v již zmíněném návodu „Sdílení tisku mezi systémy Windows a Debian“.

Tisk z Apple

Netatalk podporuje tisk klientů Apple protokolem EtherTalk. Další informace najdete v dokumentu

Netatalk HOWTO, *http://thehamptons.com/anders/netatalk/*.Modernější verze Macu dokážou tisknout i přes TCP/IP protokolem LPD. UV nabízí velmi pěknoustránku *http://www.itc.virginia.edu/desktop/mac/ip_printing/ip_printing.html*, kde se dozvíte po-drobnosti o nastavení.

Tisk z NetWare

Balík ncpfs obsahuje démona pserver, který umí poskytovat službu tiskového serveru frontám NetWare. Pokud vím, tento systém vyžaduje NetWare založený na Bindery, tedy NetWare 2.x, 3.x nebo 4.x s aktivací přístupu k bindery.

Další informace o ncpfs a také o programu pserver najdete na FTP stránkách ncpfs (ftp://ftp.gwdg.de/pub/linux/misc/ncpfs/).

Administrace síových tiskáren

Většina síťových tiskáren podporuje nějakou metodu vzdálené správy. Často jde o snadno použi-telné webové rozhraní. Ještě užitečnější je podpora správy přes SNMP. Typicky tak můžete zjistit zajímavé informace o stavu tiskárny, jako jsou množství toneru a papíru, o zatížení tiskárny a můžete také měnit některá nastavení. Ovládání tiskáren přes SNMP a řada dalších věcí týkajících se tisku je standardizována skupinou Printer Working Group pod IEEE, *http://www.pwg.org/*.

npadmin

Npadmin (*http://npadmin.sourceforge.net/*) je řádkový program poskytující rozhraní k běžným SNMP funkcím síťových tiskáren. Implementuje standard Printer MIB (*http://www.ietf.org/rfc/ rfc1759.txt*) a některá proprietární řešení výrobců, která se týkají zejména starších zařízení. Pod-porovány jsou jak funkce printer-discovery, tak i různé stavové dotazy na tiskárny.

Program npadmin má vynikající manuálovou stránku (*http://npadmin.sourceforge.net/man/*) a existuje i v balíčcích RPM a dpkg pro distribuce, které tento typ balíčků používají.

Další SNMP nástroje

Kromě npadmin existuje i několik dalších užitečných SNMP nástrojů: snmptraplogd může logo-vat události SNMP trapů. Je to užitečné pro sledování zaseknutí papíru, vyčerpání papíru a podob-ně, jednoduché je také přesměrování konkrétních událostí na e-mail a podobně.

I když npadmin nabízí jednoduchou podporu SNMP rozhraní většiny síťových tiskáren, některé tiskárny používají i různá proprietární rozšíření, která npadmin nezná. V takovém případě může-te použít nástroje CMU SNMP, které podporují libovolné operace SNMP GET a SET. S trochou práce tak budete moci využít všechny funkce, které SNMP vaší tiskárny nabízí. Budete potřebovat specifikaci výrobce, abyste zjistili, co všechny proměnné znamenají – výrobci často předpokláda-jí, že uživatelé používají výhradně jimi dodávané proprietární nástroje.

Knihovna libprinterconf (*http://sourceforge.net/project/?group_id=3648*) VA Linuxu nabízí funkci detekce síťových tiskáren. Identifikace tiskáren probíhá podle vestavěné databáze signatur tiská-ren. Tato databáze není momentálně příliš rozsáhlá, pokrývá ale většinu běžných modelů.

Jak vytvořit něco, co stojí za vytisknutí

Teď už se dostáváme k poslednímu kroku v softwarovém řetězci. V zásadě Linux umí spustit řadu typů binárních souborů s různým stupněm úspěšnosti: Linux/x86, Linux/Alpha, Linux/Sparc, Linux/foo, iBCS, Win16/Win32s (pomocí dosemu a Wine),

Mac/68k (pomocí Executor) a Java. My budeme hovořit pouze o nativním Linuxu a běžném unixovém softwaru.

Značkovací jazyky

Většina značkovacích (mark-up) jazyků se hodí zejména pro větší nebo opakující se projekty, kdy počítač upravuje pomocí značek vzhled textu tak, aby vypadal jednotně.

nroff

Jde o jeden z prvních značkovacích jazyků na původní verzi Unixu. Nejznámějším příkladem věcí formátovaných pomocí maker *roff jsou manuálové stránky. Řada lidí na ně nedá dopus-tit, nicméně alespoň podle mě má nroff příliš záhadnou syntaxi (viz obrázek 11.14) a v sou-časné době nepředstavuje nejlepší volbu. Je ale dobré vědět, že manuálovou stránku můžete vysázet přímo v PostScriptu pomocí programu groff. Většina příkazů man to umí sama pří-kazem man -t foo | lpr.

```
.B man is the system's manual pager. Each .I page argument given to .B man is normally the name of a program, utility or function. The .I manual
page associated with each of these arguments is then found and displayed. A .IR section , if provided, will direct .B man to look only in that .I
section of the manual.
```

Příklad vstupu pro roff

TeX

TeX a balík maker LaTeX představují jeden z nejrozšířenějších značkovacích jazyků na unixo-vých systémech, i když TeX nepochází původně z Unixu a je k dispozici na řadě různých systé-mů. V LaTeXu se velmi často vytvářejí technické dokumenty, protože je velmi zjednodušena problematika sazby a LaTeX je navíc jeden z mála systémů, který podporuje sazbu matema-tických vzorců jak kompletně, tak i dobře. Výstupem TeXu je formát *dvi*, který je možné pro-gramy dvips a dvilj konvertovat na PostScript nebo PCL. Pokud budete chtít nainstalovat TeX nebo LaTeX, nainstalujte celou skupinu balíků teTeX, která obsahuje všechno. Nové verze TeXu obsahují i pdfTeX a pdfLaTeX, které vytvářejí přímo soubory PDF. Více o TeXu najdete např. na stránkách *http://www.cstug.cz/*.

K dispozici jsou i příkazy pro vkládání odkazů a navigačních funkcí do PDF dokumentů.

SGML

Pro unixové systémy existuje přinejmenším jeden volně šířený parser SGML, díky němuž vznikl dokumentační systém Linuxdoc, založený na SGML. Může podporovat i jiné DTD, zejména DocBook. Tento dokument byl napsán v DocBook-DTD SGML, příklad vidíte na následujícím obrázku.

```
\subsubsection{NAT}

Each real server is assigned a different IP address, and the NA implements address translation for all inbound and outbound packets.

\begin{description} \item[Advantage] Implementation simplicity, especially if we already implement other
NAT capabilities.

  \item[Disadvantage] Return traffic from the server goes through address translation, which may incur a speed penalty. This probably isn't too
bad if we design for it from the beginning.

  \item[Disadvantage] NAT breaks the end-to-end semantics of normal internet traffic. Protocols like ftp, H.323, etc would require special
support involving snooping and in-stream rewriting, or complete protocol proxying; neither is likely to be practical.
  \end{description}
```

Příklad vstupu pro LaTeX

```
<varlistentry><term>SGML</term><listitem><para>There is at least one free SGML parser available for Un*x systems; it forms the basis of
Linuxdoc-SGML's homegrown document system. It can support other DTD's, as well, mostnotably DocBook. This document is written in
DocBook-DTD SGML.</para></listitem></varlistentry>
```

Příklad DocBook SGML

Textové WYSIWYG procesory

Programů pro WYSIWYG práci s textem je dostatek. Existují úplné kancelářské balíky, jeden z nich je pro osobní potřebu zdarma (StarOffice).

StarOffice/OpenOffice.org

Sun Microsystems distribuoval původní StarOffice pro Linux zdarma, ale následně jej přejme-noval na OpenOffice, otevřel vývoj a uvolnil zdrojový kód pod open-source licencí. Dnes si můžete stáhnout kompletní kancelářský balík např. z adresy

http://www.openoffice.cz. Sun tento kancelářský balík pod názvem StarOffice stále prodává. V podstatě jde o upravený Open-Office.org s přidanými funkcemi, podporou atd. StarOffice/OpenOffice.org je kompletní kan-celářský balík obsahující všechny funkce, které byste očekávali, včetně importu a exportu dokumentů ve formátu Microsoft Office (tedy Word a podobně). Existuje mini-HOWTO doku-ment popisující získání a instalaci tohoto balíku. Tiskovým výstupem je PostScript, takže by měl fungovat s jakoukoliv tiskárnou, která v Linuxu funguje. Na OpenOffice.org je postaven i český produkt 602Office.

WordPerfect

Corel distribuuje základní verzi WordPerfectu 8 pro Linux zdarma a prodává různé balíky Word Perfect Office 2000 (obsahující WordPerfect, Corel Draw a Quatro Pro verze 9). Stránka *Linux WordPerfect Fonts and Printers* (*http://www.rodsbooks*) obsahuje informace o konfigu-raci WordPerfectu pro použití s Ghostscriptem nebo s originálními ovladači WordPerfectu (které jsou stejné jako ovladače pro DOS pro případ, že by ovladač vaší tiskárny v distribuci nebyl).

Applix

Applix je multiplatformní (Unixy, Windows a další) kancelářský balík prodávaný společností Applix. Red Hat a SuSE jej prodávaly rovněž v době, kdy neexistovala jiná varianta, nyní jej prodává už zase jen Applix. Jedná se o jediný aplikační balík v nativním unixovém stylu, prav-děpodobně se vám bude líbit jeho unixový přístup k věci.

AbiWord

AbiWord, *http://www.abisource.com/*, je jeden z několika GPL WYSIWYG editorů. Jde o velmi pěkný editor založený na formátu XML. Podporuje také import souborů z Wordu. Vývoj Abi-Wordu stále pokračuje, už dnes je ale velmi dobře použitelný.

LyX

LyX je rozhraní k LaTeXu, které vypadá velmi slibně. Další informace najdete na adrese *http://www.lyx.org/.* Existuje i KDE verze LyXu, pojmenovaná Klyx.

Maxwell

Maxwell je jednoduchý textový editor založený na formátu MS RTF, který začal jako komerční produkt a nyní se distribuuje pod GPL.

KOffice

Moderní kancelářský balík integrovaný s prostředím KDE, více na *http://www.koffice.org.* Pod-poruje formát Open Document – nativní formát balíku OpenOffice.org.

Uvítám informace o dalších produktech.

Tisk fotografií

Abyste z normálních tiskáren získali slušné fotografie, je nutné uhlídat celou řadu detailů. Pokud ještě nemáte fototiskárnu, podívejte se na tipy v kapitole „Jak kupovat tiskárnu“.

Ghostscript a fotografie

Ghostscript má potíže s tiskem barevných fotografií u většiny ovladačů. Problémy spočívají v několika věcech:

- Řada ovladačů má špatně vyladěnou podporu barev. Barvy často neodpovídají podobě na obrazovce a výtisku ve Windows. Na druhé straně, všechny ovladače i samotný Ghost script mají nastavitelnou barevnou podporu. Jednou z možností je nastavení gama korekce (viz kapitola „Gamma, velikost bodu atd.“), další jsou popsány v dokumentačním souboru Use.htm Ghostscriptu.

Vím pouze o jediném ovladači pro Ghostscript, který podporuje šesti a sedmibarevný tisk, momentálně je v beta-verzi a podporuje modely Epson Stylus Photo. Tvrdí se, že barevný výstup je lepší než u originálního ovladače pro Windows. Jádro Ghostscriptu nepodporu-je jiné barevné modely než CMYK a RGB, je nutné tuto podporu dopracovat.
Ghostscript často generuje nehezký dithering a vytváří různé chyby, například proužky. Dithering lze obvykle opravit, viz kapitola „Barevný tisk v Ghostscripu“ a dokumentace k příslušnému ovladači.

Některé z těchto problémů lze upravit doladěním Ghostscriptu. Podrobnosti najdete v kapitole „Ghostscript“. Nastavování parametrů Ghostscriptu se výrazně zjednoduší, pokud je deklarujete jako parametry tiskového systému.

Momentálně je tedy lepší fotografie tisknout jiným systémem než Ghostscriptem, takové progra-my samozřejmě existují. Nejvýznamnějším je tiskový doplněk pro GIMP, který podporuje tisk na Epson Stylus a na postscriptových tiskárnách (s podporou PPD). Část podporující Epson Stylus existuje i pro Ghostscript jako ovladač stp. Další variantou jsou externí programy

pnm-to-něco, podporující tisk na tiskárnách, jako je Lexmark – tyto programy provádějí tisk mapování pixel-napixel. Nejlepší řešení je samozřejmě koupě postscriptové tiskárny, ty lze obvykle plně ovládat dostupnými programy a při tisku využít všech možností tiskárny, případně pořízení tiskárny s kva-litním nativním ovladačem pro Linux.

Papír

U barevných inkoustových tiskáren závisí kvalita tisku významně na použitém papíru. Drahé lesk-lé papíry umožňují tisknout v téměř fotografické kvalitě, výstup na klasický kancelářský papír dává obvykle ušmudlané barvy a rozplizlé detaily. Speciální matné papíry do inkoustových tiskáren dávají výsledek někde uprostřed a hodí se zejména pro tisk finálních podob dokumentů. Tvrdé lesklé fotografické papíry dávají stejný výsledek jako „normální“ lesklé fotopapíry, výsledek však více připomíná klasickou fotografii.

Nastavení tiskárny

Při tisku fotografií byste na většině barevných inkoustových tiskáren měli použít nejkvalitnější (a nejpomalejší) režim tisku, jinak budou větší plochy pruhované nebo barevně nejasné. V Ghost-scriptu obvykle stačí zvolit nejvyšší rozlišení. U postscriptových tiskáren budete možná muset před samotnou úlohou tiskárnu nastavit nějakým příkazem podle PPD souboru. PPD podpora v GIMP neobsahuje nastavení kvality tisku (rozuměj specifické nastavení konkrétní tiskárny), nicméně pro vlastní potřebu jsem si tuto podporu doplnil – pokud chcete, kontaktujte mne. Pokud používáte PDQ nebo CUPS, budete moci snadno nastavovat všechny parametry tiskárny.

U postscriptových tiskáren nabízí tato nastavení i libppd VA Linuxu a GPR.

Trvanlivost tisku

Barevné výtisky z inkoustových tiskáren obvykle za pár let vyblednou, zejména pokud jsou vysta-veny slunečnímu světlu – to je vlastnost inkoustu. Tiskárny s výměnnými inkoustovými náplněmi, jako Epson nebo Canon, mohou pracovat i s takzvaným archivním inkoustem, který je na bled-nutí méně náchylný. Novější tiskárny obvykle používají pigmentové inkousty, které neblednou tak rychle jako starší barvené inkousty. Nicméně pro delší skladování se inkoustové výtisky obecně nehodí – vypalte si obrázky na CD a uskladněte je takto.

Komerční programy

Existuje program *xwtools*, *http://home.t-online.de/home/jj.sarton/startE.htm*, který umožňuje tisk fotografií se všemi parádičkami na vybraných tiskárnách Epson, HP a Canon. Bohužel byl napsán pod NDA, takže nejsou k dispozici jeho zdrojové kódy.

Balík ESP Print Pro společnosti Easy Software podporuje některé jinak nepodporované tiskárny.

Tyto ovladače nejsou ideální pro tisk fotografií, ale aspoň fungují.Tisk fotografií lze vylepšit také použitím jiných ovladačů, za všechny jmenujme například komerč-ní Turboprint, *http://www.turboprint.de.*

Další informace

Tiskárny výlučně pro Windows

Jak už bylo řečeno dříve, některé tiskárny jsou ze své podstaty nepodporované, protože neznají žádný „normální“ komunikační jazyk, namísto toho využívají procesoru počítače k vykreslení tis-kového rastru, který se pak fixní rychlostí posílá na tiskárny. V některých případech tyto tiskárny umějí i něco normálního jako PCL, ale často ani to ne. U některých (opravdu „low-end“ modelů) tyto tiskárny dokonce ani nepoužívají běžnou paralelní komunikaci, ale spoléhají na to, že ovla-dač emuluje to, co má řešit hardware (například řízení komunikace).

Každopádně, pokud na nějakou takovou hrůzu narazíte, existuje několik možných řešení.

Redirektor Ghostscriptu

V současné době existuje tiskový ovladač pro Ghostscript (jmenuje se mswinpr2), který tiskne pomocí GDI volání Windows. Existuje také nástroj pro přesměrování portu, redmon, který umož-ňuje úlohu před konečným tiskem prohnat přes Ghostscript (podobně jako to dělá filtr if v LPD v Unixu). Díky tomu mohou Windows tisknout PostScript s použitím originálního ovladače.

Pokud na tiskárně nemůžete tisknout přímo, můžete ji exportovat jako postscriptovou tiskárnu pomocí programů redmon, Ghostscript a mswinpr2 ve Windows a tisknout pomocí ovladače od výrobce.

HP Winprinters

Některé tiskárny HP používají „Printing Performance Architecture“ (což je marketingový výraz pro „ta tiskárna je moc levná, než aby uměla PCL“). Tato metoda je podporována překladačem pbm2ppa Tima Normana. V zásadě použijete Ghostscript k vykreslení PostScriptu do rastrového obrázku ve formátu pbm a pak použijete pbm2ppa ke konverzi tohoto rastru do interního rastru tiskárny, který je možné vytisknout. V současné době už by měl být tento program k dispozici i jako ovladač pro

Ghostscript.

Tento program můžete získat na adrese *http://www.rpi.edu/~normat/technical/ppa/*, pbm2ppa podporuje některé modely HP 720, 820 a 1000. Podrobnosti se dozvíte v dokumentaci k tomuto balíku. Nové tiskárny HP jsou již dobře podporovány projektem HPIJS, viz *http:// hplip.sourceforge.net/supported_devices/index.html.*

Lexmark Winprinters

Většina levných inkoustových tiskáren Lexmark používá proprietární jazyk, a jsou to tedy klasic-ké Winprinters. Nicméně Henryk Paluch napsal program, který umí tisknout na tiskárně Lexmark 7000. Možná bude schopen rozšířit podporu i o barevný tisk a další tiskárny Lexmark. Další infor-mace najdete na adrese *http://bimbo.fjfi.cvut.cz/~paluch/l7kdriver/*.

Podobně existují i ovladače pro modely 5700, 1000, 1100, 2070, 3200 a další. Podívejte se do data-báze podporovaných tiskáren, kde najdete i odkazy na příslušné ovladače.

Jak tisknout na fax

Na fax můžete tisknout s pomocí nebo bez pomoci modemu.

Použití faxmodemu

Existuje řada faxovacích programů, které umožňují faxovat a přijímat dokumenty. Jedním z nej-lepších je HylaFAX Sama Lefflera, *http://www.hylafax.org/.* Podporuje všechno možné, počínaje více modemy až po broadcasting.

SuSE obsahuje klienta HylaFAX napsaného v Javě, který údajně pracuje na jakékoliv platformě s podporou Javy (včetně Windows a Linuxu). Kromě toho pro většinu platforem existují i faxoví klienti nepoužívající Javu, v Linuxu bude téměř jisté nastavit faxování podle vašich potřeb.

Další možností, vhodnější pro malé instalace, je *efax*, *http://casas.ee.ubc.ca/efax/*, program, který odesílá a přijímá faxy. Program mgetty umí spolupracovat s efaxem a přijímat faxy (a také hlaso-vou poštu a interaktivní přihlášení).

Služba Remote Printing Service

Existuje experimentální služba, kam můžete poslat e-mail obsahující to, co chcete někomu poslat na fax. Podporován je i PostScript, takže i přesto, že služba není dostupná všude, může být veli-ce užitečná. Další informace o této službě najdete na adrese *http://www.tpc.int/*.

Komerční faxové služby

Řada společností nabízí faxové služby přes Internet. Například EFax, *http://www.efax.com/*, nabí-zí zdarma příjem faxů přes e-mail a odesílání faxů za poplatek. Podobné služby nabízejí i další společnosti.

Náhled před tiskem

Téměř vše, co jde vytisknout, si můžete také prohlédnout na obrazovce.

PostScript

Ghostscript obsahuje ovladač pro X Window, který se nejlépe používá s prohlížečem PostScriptu gv (*http://wwwthep.physik.uni-mainz.de/~plass/gv/*). Nové verze tohoto programu umí zobrazit i soubory PDF. Program gv je náhradou staršího prohlížeče Ghostview, nové rozhraní je mnohem pěknější a chytřejší než původní.

Tex dvi

Soubory DeVice Independent TeXu si můžete v X Window prohlížet pomocí xdvi, *http://www.linuxprinting.org/man/xdvi.1.html.* Moderní verze xdvi volají pro vykreslení specialit PostScriptu Ghostscript.

Existuje také ovladač pro VT100, jmenuje se dgvt. Prohlížeč tmview používá knihovnu svgalib, pokud nemáte nic lepšího k dispozici.

Adobe Reader

Adobe Reader existuje i pro Linux, můžete si jej stáhnout na *http://www.adobe.com/.*
Kromě toho můžete použít xpdf, což je volně šiřitelný software a bývá součástí distribucí. I gv podporuje prohlížení PDF souborů pomocí gs pod X Window, viz například předchozí obrázek.

Sériové tiskárny a LPD

Použití sériových tiskáren s lpd je trochu složité.

Nastavení v printcap

LPD nabízí pět parametrů, kterými můžete v /etc/printcap nastavit všechny vlastnosti sériové-ho portu a připojení tiskárny. Přečtěte si manuálovou stránku programu printcap(*http://www.linuxprinting.org/man/printcap.5.html*) a popis parametrů br#, fc#, xc#, fs# a xs#.Poslední čtyři představují bitové mapy sloužící k nastavení portu, parametr br# nastavuje přeno-sovou rychlost, například br#9600.

Velmi jednoduchý je převod nastavení stty (*http://www.linuxprinting.org/man/stty.1.html*) na pří-
znaky pro printcap. Podívejte se na manuálovou stránku stty.Pomocí stty nastavte port tiskárny tak, abyste byli schopni příkazem cat na tiskárnu poslata vytisknout soubor. Takto vypadá příkaz stty -a pro můj port tiskárny:

dina:/usr/users/andy/work/lpd/lpd# stty -a < /dev/ttyS2 speed 9600 baud; rows 0; columns 0; line = 0; intr = ^C; quit = ^\; erase = ^?; kill = ^U; eof = ^D; eol = <undef>; eol2 = <undef>; start = ^Q; stop = ^S; susp = ^Z; rprnt = ^R; werase = ^W; lnext = ^V; min = 1; time = 0; -parenb -parodd cs8 hupcl -cstopb cread -clocal -crtscts -ignbrk -brkint -ignpar -parmrk -inpck -istrip -inlcr -igncr -icrnl ixon -ixoff -iuclc -ixany -imaxbel -opost -olcuc -ocrnl -onlcr -onocr -onlret -ofill -ofdel nl0 cr0 tab0 bs0 vt0 ff0 -isig -icanon -iexten -echo -echoe -echok -echonl -noflsh -xcase -tostop -echoprt -echoctl -echoke

Jediné rozdíly tohoto nastavení proti nastavení při startu systému jsou příznaky -clocal, -crtscts a ixon. Možná budete potřebovat úplně jiná nastavení v závislosti na tom, jaké používá tiskárna řízení přenosu.

Ve skutečnosti musíte stty použít poněkud zvláštním způsobem. Protože stty pracuje s terminá-lem připojeným na standardní vstup, manipulujete se sériovým portem pomocí znaku „<“ tak, jak to vidíte výše.

Jakmile máte stty nastaveno správně, takže příkaz cat file > /dev/ttyS2 (v mém případě) pošle soubor na tiskárnu, podívejte se do souboru /usr/src/linux/include/asm-i386/termbits.h. Tento soubor obsahuje řadu příkazů #define a několik struktur. (Případně můžete tento soubor poslat na tiskárnu (to vám funguje, že?) a použít jej jako referenci.) Najděte si část, která začíná:

/* c_cflag bit meaning */ #define CBAUD 0000017

V této části najdete význam bitů pro parametry fc# a fs#. Všimněte si, že názvy v tomto soubo-ru odpovídají názvům, které se objevují ve výstupu příkazu stty. Neříkal jsem, že to bude jedno-duché?

Všimněte si, která nastavení jsou ve výstupu příkazu stty uvedena znakem „-“. Sečtěte hodnoty u všech těchto parametrů (hodnoty jsou v osmičkové soustavě). Tím dostáváte bity, které musíte vynulovat, a tato hodnota slouží jako argument parametru fc#. Nicméně hned vzápětí budeme
nastavovat potřebné bity, takže nejjednodušší je prostě nejprve vynulovat všechno, což můžete

provést volbou fc#0177777 (já to tak dělám).Teď sečtěte hodnoty u parametrů, u nichž není uveden na začátku výpisu příkazu stty znak -.V mém případě jde o parametry CS8 (0000060), HUPCL (0002000) a CREAD (0000200). Dále neza-pomeňte na příznaky nastavující přenosovou rychlost (u mne je to 0000015). Všechno sečtěte,v mém případě dostanete hodnotu 0002275. To je argument pro parametr fs# (v mém případěstačí zadat fs#02275).

Stejný postup s nulováním a nastavováním bitů proveďte i s další částí souboru, c_lflag. V mémpřípadě se nenastavuje nic, takže používám parametry xc#0157777 a xs#0.

Starší sériové tiskárny vynechávající znaky

Jon Luckey upozornil, že některé starší sériové tiskárny s levným sériovým rozhraním a malou vyrovnávací pamětí opravdu myslí STOP, když to v řízení přenosu signalizují. Zjistil, že pokud pří-kazem setserial (*http://www.linuxprinting.org/man/setserial.8.html*) vypnul frontu u sériového portu s čipem 16550, vyřešil se problém s vynecháváním znaků. (Stačí pouze označit typ uart jako 8250.)

Co chybí?

Ještě stále chybí řada částí do úplného tiskového systému. Existují projekty, které se je snaží dopl-nit, ovšem většina z nich doposud nic rozumného nevyřešila a snahy o standardizaci potřebných protokolů a API jsou stále v plenkách, ale situace se postupně zlepšuje. Do snahy o standardiza-ci se dnes zapojují velcí linuxoví hráči i firmy vyrábějící tiskárny, viz *http://www.linux-foundation.org/en/OpenPrinting.*

Propojení

Existuje obecný problém donutit všechny části, aby spolu navzájem komunikovaly, zejména mechanismem nezávislým na použitém spooleru. Tento problém se nejzřetelněji projevuje u dojemné snahy aplikací řídit všechny „obvyklé“ tiskové funkce. Autor aplikace prostě nemá možnost zjistit informace o tiskárně, úlohách a podobně, neexistuje standardizovaná metoda vytvoření úlohy, žádný rozumný způsob, jak se dozvědět stav úlohy, dokonce ani žádný stan-dardní způsob generování tiskových dat (i když většina moderních systémů nabízí nějaká speci-fická řešení).

Fonty

Obsluha fontů je hrozně nešikovná. Obrazovka, tiskárna, aplikace a datové soubory by měly mít v ideálním případě přístup ke stejným fontům. To však, bohužel, není pravda. S příchodem xft2 a fontconfig – což začnou šířit nejnovější distribuce – by to mělo být vyřešeno. Pokud vím, Red-hat 8.0 je první distribuce, která používá xft2. V současné době jej používají snad všechny

moder-ní distribuce a podpora práce s fonty se razantně zlepšila. Více o fontech najdete v návodu „Opti-mální použití písem v systému Linux“.

Ovladače

Stav na poli volně šířených ovladačů je poměrně bídný. I když se za posledních pár let dost vylep

šily, ne všechny tiskárny jsou podporovány.
V této oblasti hrají hlavní roli prodejci tiskáren. S rostoucí oblibou Linuxu bude pro ně čím dál tím těžší ignorovat tuto skupinu uživatelů. Většina výrobců dnes už nabízí nativní ovladače pro Linux, nejlepší podporu mají pravděpodobně tiskárny HP díky projektu HPIJS.

Sdílení tisku mezi systémy Windows a Debian

Úvod

Debian GNU/Linux (*http://www.debian.org*) patří k předním distribucím udržovanou dobrovolní-ky Linuxu. Bohužel může být nastavení tiskáren v Debianu obtížné. Snadné není ani vyhledání instrukcí pro sdílení tiskáren mezi systémy Windows a Linux za použití nejnovějších nástrojů. Oba problémy se snaží rychle a jednoduše vyřešit tento dokument.

Dozvíte se, jak používat nástroje příkazového řádku při konfiguraci systému Debian pro tisk. Zjistíte, jak odesílat dokumenty ze systému Linux na tiskárny systému Windows a jakým způsobem lze sdílet tiskárny Linux s počítači Windows. Ukážeme si též řešení některých potíží. Tento návod je zaměřen ryze prakticky, komplexní informace o tiskovém systému v Linuxu najdete v návodu „Tisk v Linuxu“ dále v této knize. Nutno také dodat, že v jiných distribucích bude postup pravděpodobně velmi podobný.

Začínáme

Součásti tisku v systému Linux

Používat budeme hlavně následující součásti:

CUPS

Common UNIX Printing System (*http://www.cups.org*) je systém pro zpracování tisku v Linuxu a sada podpůrných programů pro používání a správu tiskáren.

Samba

Samba (*http://www.samba.org*) je software, který umožňuje počítačům s jiným systémem než Win-dows, aby na síti vypadaly jako systém Windows díky implementaci protokolu pro sdílení tiská-ren a souborů.

Ovladače tiskárny

Web LinuxPrinting.org (*http://www.linuxprinting.org*) nabízí obrovský počet ovladačů tiskáren a nabízí databázi tiskáren podporovaných v systému Linux. Pro každý model tiskárny, který chce-te používat v systému Linux, musíte stáhnout příslušný ovladač. Ovladač tiskárny se skládá ze sou-boru PPD a filtrovacího programu, případně jde jen o soubor PPD pro tiskárny PostScript.

Vyžadované balíčky

Součástí standardního archívu Debian jsou všechny požadované programy a knihovny. Tyto balíč-ky si můžete stáhnout a nainstalovat za použití běžných nástrojů Debianu. Potřebovat budete následující balíčky:

cupsys

Tiskový server pro systém CUPS.

cupsys-bsd

Příkazy CUPS BSD.

cupsys-client

Klientské programy pro CUPS.

foomatic-bin

Podpůrné programy pro tisk z webu LinuxPrinting.org.

samba

Server Samba SMB/CIFS pro UNIX.

smbclient

Klient Samba SMB/CIFS pro UNIX.

gs-esp

ESP Ghostscript (*http://www.cups.org/ghostscript.php*). Není k dispozici jako balíček pro Debian verze 3.0 (známý také jako Woody), použijte místo něj balíček gs.

a2ps

Nástroje GNU A2PS (*http://www.gnu.org/software/a2ps/*).Tyto balíčky nainstalujete pomocí následujících příkazů. Pro spuštění těchto příkazů se musíte při-hlásit jako root nebo použít sudo:

apt-get update apt-get install cupsys cupsys-bsd cupsys-client foomatic-bin samba smbclient gs-esp a2ps

Některé tiskárny mohou vyžadovat další balíčky. Například pro správné fungování mnoha tiská-ren HP InkJet, DeskJet a LaserJet je nutné nainstalovat balíček *hpijs*. Soubory PPD pro tyto tiskár-ny jsou označeny řetězcem „hpijs“ v názvu souboru.

Nastavení lokální tiskárny

Pro konfiguraci tiskáren se používá příkaz lpadmin. V následujícím příkladu nastavíte laserovou tiskárnu prostřednictvím CUPS. Pro spuštění těchto příkazů se musíte přihlásit jako root nebo pou-žít sudo:

/usr/sbin/lpadmin -p Laser -v parallel:/dev/lp0 -P /root/laser.ppd /usr/bin/enable Laser /usr/sbin/accept Laser /usr/sbin/lpadmin -d Laser

Měli byste vědět, že bash má vestavěný příkaz s názvem enable (viz část „Bash pro začátečníky“), takže uživatelé bashe *musí* pro povolení tiskáren *použít úplnou cestu* (/usr/bin/enable). První příkaz vytvoří novou tiskárnu s názvem „Laser“, která je připojená k prvnímu paralelnímu portu a používá soubor PPD /root/laser.ppd. Poté je tiskárna „Laser“ povolena prostřednictvím příkazu enable a příkazem accept zajistíte příjem tiskových úloh. Poslední příkaz nastavuje „Laser“ jako výchozí tiskárnu.

Je-li tiskárna připojena k portu USB nebo neznáte správnou konfiguraci, zkuste použít příkaz /usr/sbin/lpinfo -v pro zobrazení seznamu dostupných tiskáren. Spuštěním příkazu /usr/bin/lpoptions -l se ujistěte, zda je správně nastavena velikost papírua další možnosti pro tiskárnu. Podrobnější informace o konfiguraci tiskárny jsou k dispoziciv dokumentaci CUPS.

Základy tisku v systému Linux

Dokumenty se zpracovávají buď příkazem lpr nebo lp, za kterým následuje název souboru. Fron-tu a stav tiskárny můžete zobrazit pomocí příkazu lpstat -o či lpstat -p. Tiskovou úlohu můžete zrušit příkazem cancel nebo lprm, za kterým následuje identifikátor úlohy.

Démon systému CUPS se nazývá *cupsd*. Převádí dokumenty na PostScript a poté je převádí na for-mát, který je nativní pro tiskárnu (viz obrázek 12.1). Tiskárny, které nerozumí formátu PostScript, používají pro dokumenty rastrový formát. Rastrové formáty mohou být mnohem větší než původ-ní PostScript a jejich odeslání na tiskárnu bude trvat déle.

Filtry jsou programy, které slouží k převádění dokumentů z jednoho formátu na jiný. Program CUPS se bude snažit nalézt nejvhodnější filtr pro vámi odeslané dokumenty. Pokud není nainsta-lovaný žádný formát vhodný pro převod dokumentu, zobrazí se chybová zpráva podobná této: „lpr: unable to print file: client-error-document-format-not-supported.“

Mnoho aplikací neobsahuje filtry pro vlastní formáty dokumentů. Dokumenty vytvořené v těchto aplikacích lze tisknout pouze v rámci dané aplikace, nikoliv standardními tiskovými příkazy. Případně musí být dokument exportován do formátu PostScript či jiného standardního tiskového formátu.

Tisk na počítače s Windows

Připojení k Windows

Nativním protokolem Windows pro sdílení souborů a tiskáren je SMB (CIFS). Při komunikaci s Windows počítači používáme software Samba, který těmto protokolům rozumí, viz obrázek 12.2. Před nastavením systému CUPS na tisk do Windows se pomocí programu smbclient přesvědčte, že se můžete připojit k počítači s Windows.

Následující příklad ukazuje, jak se spojit s počítačem, na kterém běží Windows:

/usr/bin/smbclient -L rice -U fred

added interface ip=10.6.7.234 bcast=10.6.7.255 nmask=255.255.255.0 Got a positive name query response from 10.6.7.8 (10.6.7.8) Password: (not shown)

```
Sharename       Type      Comment
PRINTER$        Disk
INKJET                                  Printer
STUFF           Disk
IPC$            IPC       Remote Inter Process Communication
```

Příkaz smbclient se zeptal počítače s názvem „rice“, jaké jsou jeho sdílené prostředky, a použí-val přitom identitu uživatele „fred“. Na výsledku vidíte, že počítač s Windows sdílí tiskárnu s názvem INKJET.

Není-li dostupná názvová služba Windows, nelze použít jméno a musíte počítač s Windows iden-tifikovat pomocí jeho IP adresy a parametru -I například takto:

```
/usr/bin/smbclient -I 10.6.7.8 -L rice -N
```

Více informací o použití programu smbclient najdete v dokumentaci k Sambě.

Nastavení systému CUPS

Po nalezení sdílené tiskárny Windows můžete nastavit CUPS tak, aby ji začal používat. Nejdříve zkontrolujte, zda má vaše instalace systému CUPS k dispozici tiskovou komponentu podporující protokol SMB pomocí následujícího příkazu:

```
ls -l /usr/lib/cups/backend/smb
```

Jestliže tento soubor neexistuje, vyrobte ho následujícím příkazem:

```
ln -s `which smbspool` /usr/lib/cups/backend/smb
```

Dále si ukážeme přidání tiskárny, kterou jsme našli v předchozí kapitole. Všechny následující kroky budete muset provést buď jako uživatel root nebo pomocí příkazu sudo:

```
/usr/sbin/lpadmin -p RicePrinter -v smb://fred:mypass@rice/INKJET -P /root/inkjet.ppd /usr/bin/enable RicePrinter /usr/sbin/accept
RicePrinter /usr/sbin/lpadmin -d RicePrinter
```

Jak jsme již říkali, uživatelé příkazového interpretru bash musí pro povolení tiskáren použít úpl
nou cestu k programu enable (/usr/bin/enable) z důvodu kolize s interním příkazem bashe. Příkaz lpadmin nastaví v Linuxu tiskárnu s názvem „RicePrinter“ sdílenou ze systému Windows. Použije přitom její název na vzdáleném počítači a dále netbios (windows) název počítače „rice“ a uživatele „fred“ s heslem „mypass“. Následně tiskárnu povolí a nastaví ji jako výchozí tiskárnu v sytému, viz příklad v kapitole „Nastavení lokální tiskárny“.

Tiskárna je připravena k použití a můžete si ji otestovat. Zkuste nyní vytisknout nějaký soubor přímo pomocí příkazu lp /cesta/k/souboru, nikoliv z aplikace.

Sdílení tiskáren s počítači Windows

Základy sdílení

Samba používá pro sdílení souborů a tiskáren s počítači Windows démony *nmbd* a *smbd*. První z nich – nmbd – funguje jako názvová služba Windows, přičemž vysílá název vašeho počítače do systémů Windows v místní síti LAN. Démon smbd přijímá soubory a požadavky na tisk z počíta-čů Windows (viz obrázek 12.3).

Ovladače tiskáren Windows budete muset stáhnout a nainstalovat pro každou tiskárnu sdílenou v systému Linux. Ovladače tiskáren Windows naleznete na webu výrobce dané tiskárny.

Konfigurace Samby

Jestliže chcete povolit anonymní přístup ke své tiskárně, budete muset vytvořit uživatelský účet pro vzdálené tiskové úlohy:

```
/usr/sbin/adduser --system --disabled-password smbprint
```

Tento příkaz přidá do vašeho systému uživatele s názvem „smbprint“. Ujistěte se, že v domovském adresáři uživatele /home/smbprint je dostatek prostoru pro soubory programu pro zpracování tisku. Zkontrolujte, že uživatel smbprint nemá ve vašem systému oprávnění pro čtení či úpravu důležitých souborů a adresářů. Pokud jste nakonfigurovali CUPS pro zamezení tisku pro určité uži-vatele, musíte povolit uživateli smbprint přístup k tiskárnám, které chcete sdílet.

Konfiguračním souborem Samba je /etc/samba/smb.conf. V následujícím příkladu nastavujeme konfigurační soubor pro použití CUPS s uživatelem smbprint:

```
[global]
   printcap name = cups
```

printing = cups
security = share

[printers] browseable = yes printable = yes public = yes create mode = 0700 guest only = yes use client driver = yes guest account = smbprint path = /home/smbprint

Tato konfigurace umožní tisk komukoli, kdo se může prostřednictvím sítě připojit k vašemu počí-tači, a nedoporučuje se pro počítače v nedůvěryhodných sítích, například s přímým připojením k Internetu. Potřebujete-li implementovat řízení přístupu, nastavte security = user nebo secu-rity = domain a přečtěte si další informace v dokumentaci Samby.

Po přidání výše uvedených nastavení do konfiguračního souboru Samba musíte program Samba restartovat pomocí příkazu:

/etc/init.d/samba restart

Konfigurace tiskového systému CUPS

Ovladače tiskáren Windows formátují výstup pro tiskárnu před jeho odesláním do sítě. CUPS je nutné nakonfigurovat, aby bylo možné přijmout naformátovaný výstup. Učiníte tak změnou násle-dujícího řádku z /etc/cups/mime.convs:

application/octet-stream application/vnd.cups-raw 0 -

Změňte také tento řádek z /etc/cups/mime.types:

application/octet-stream

Nyní musíte CUPS nastavit tak, aby byla povolena připojení
z jiných počítačů v síti. Přidejte násle-dující řádky do
/etc/cups/cupsd.conf: <Location /printers> AuthType None Order
Deny,Allow Deny From None Allow From All </Location>

Stejně jako v konfiguraci Samba povoluje tato konfigurace připojení kteréhokoli počítače k vašim tiskárnám a nedoporučuje se pro počítače v nedůvěryhodných sítích. Informace o omezení pří-stupu k tiskárnám naleznete v souboru cupsd.conf a v dokumentaci CUPS. Nakonec restartujte cups za použití následujícího příkazu:

/etc/init.d/cupsys restart

Tiskárny Linuxu by nyní měly být nasdíleny pro počítače Windows v místní síti LAN. Proveďte obvyklé kroky pro přidání síťové tiskárny do počítačů Windows a nezapomeňte vytisknout testovací stránku.

Řešení potíží

Nedaří se připojení k tiskárnám v systému Windows

Pokud se *smbspool* (nástroj smbclient používaný v CUPS) nepodaří správně připojit k tiskárně v systému Windows, vypíše chybové zprávy, které bohužel nejsou příliš užitečné. Jednou takovou zprávou je „Unable to connect to SAMBA host: Success“. Další možnou chybou připojení je, že dokument zůstává ve frontě při tisku na tiskárnách Windows.

Prostřednictvím následujícího příkazu zobrazte poslední záznamy v logu CUPS:

/usr/bin/tail /var/log/cups/error_log

Zobrazí-li se zpráva „cli_connect() failed...“ nebo podobná, nemohl nástroj *smbspool* najít počítač Windows, ke kterému se pokoušíte připojit. Zkontrolujte správnost názvu hostitelského počítače Windows. Ověřte, zda je počítač Windows zapnutý a zda je připojení k síti funkční. Ujis-těte se, jestli je možné připojit se pomocí smbclient, jak je uvedeno v kapitole „Připojení k Windows“.

Jestliže se zobrazí zpráva „SMB tree connect failed: ERRSRV – ERRinvnetname“ nebo podobná, nástroj smbclient se připojil k počítači Windows, ale nemohl se připojit k požadované tiskárně. Zkontrolujte název sdílené tiskárny prostřednictvím smbclient, jak je uvedeno v kapito-le „Připojení k Windows“.

Jiné potíže

Může se stát, že nebudete moci tisknout na místní tiskárně a tiskové úlohy se ztrácejí z fronty, aniž by proběhl tisk. Mohou se také zobrazovat nejasné chybové zprávy, jako je například „Child pro-cess 2384 exited with status 32.“.

Rozšiřte úroveň logování systému CUPS na debug (ladění), aby se zobrazilo více zpráv o tom, co se stalo před selháním tiskové úlohy.

1. V textovém editoru otevřete hlavní konfigurační soubor CUPS /etc/cups/cupsd.conf.
2. Změňte řádek s „LogLevel warn“ na „LogLevel debug“.

Uložte konfigurační soubor a zavřete textový editor.

Restartujte server CUPS prostřednictvím příkazu:

```
/etc/init.d/cupsys restart
```

Logy tiskového systému CUPS můžete sledovat pomocí tohoto příkazu:

```
/usr/bin/tail -f /var/log/cups/error_log
```

Měli byste vidět řádek s textem „Scheduler shutting down due to SIGTERM“. To znamená, že server CUPS byl úspěšně zastaven. Znovu odešlete tiskovou úlohu a čekejte na užitečnou zprávu ladění, která se zobrazí. Příklademtakové zprávy ladění je „GNU Ghostscript 7.05: Can’t start ijs server ‘hpijs’“.V tomto případě je řešením instalace balíčku hpijs, ve kterém jsou chybějící programy.

Jestliže nemůžete určit příčinu chyby, zkuste v síti Internet vyhledat klíčové termíny obsažené ve zobrazené chybové zprávě; je pravděpodobné, že podobný problém již řešil někdo před vámi. Může-te také zkusit aktualizovat na nejnovější verze balíčky uvedené v kapitole „Vyžadované balíčky“.

Jak na spam

Úvod

Účel tohoto dokumentu

Tento dokument obsahuje informace o vysoce efektivních a neinvazivních metodách filtrování a blokování spamu a malware v průběhu příchozích transakcí protokolu SMTP na poštovním ser-veru s důrazem na odstranění zavlečeného spamu (collateral spam).

Celý text dokumentu popisuje metody obecně, součástí je však také jednoduchý příklad implementace s využitím agenta přenosu zpráv Exim a dalších specifických softwarových nástrojů.

Komu je dokument určen

Dokument je určen správcům poštovních systémů, kteří se vyznají ve zkratkách, jako je např. SMTP, MTA/MDA/MUA, DNS/rDNS nebo MX. Dokument *není* určen pro koncové uživatele, kteří hledají řešení pro filtrování spamu v prostředí svého poštovního klienta (např. Evolution, Thun-derbird, mail.app nebo Outlook Express). Takový uživatel však samozřejmě může na tento doku-ment upozornit správce své domény (firemní, školní atd.).

Nejnovější verze tohoto dokumentu

Nejnovější verzi tohoto dokumentu naleznete na adrese *http://slett.net/spam-filtering-for-mx/*. Informace na této adrese můžete kontrolovat pravidelně, dokument je průběžně aktualizován o opravy a různá doplnění.

Zpětná vazba

Rád od vás uslyším o vašich zkušenostech s technikami a nástroji, popsanými v tomto dokumen-tu, stejně jako jakékoliv jiné komentáře, dotazy, rady nebo příspěvky, které máte. Prosím zašlete mi zprávu elektronické pošty na adresu <tor na slett.net>. Pokud zvládnete implementovat tech-niky popsané v tomto dokumentu pro jiné agenty přenosu zpráv, jako je např. Sendmail nebo Postfix, dejte mi prosím vědět.

Co budete potřebovat?

Postupy popsané v tomto dokumentu vyžadují přístup k systému příchozí pošty pro doménu, do které přicházejí zprávy elektronické pošty. Kromě toho bude nutné na systému instalovat softwa-re a případně modifikovat konfigurační soubory agenta přenosu zpráv.

Diskuse v tomto dokumentu jsou obecného rázu a je možné je využít u různých agentů přenosu zpráv. Vzorová implementace je určena pro software Exim. Tato implementace zahrnuje i jiné soft-warové nástroje, jako je např. SpamAssassin. Podrobnosti implementace naleznete v dodatku A.

Poznámky k typografickým konvencím

SMTP příkazy v této kapitole vyznačujeme tučným písmem, čili např. takto: RCPT TO, EHLO. Tučně jsou vyznačeny i komunikační kódy protokolu. Části konfiguračních souborů stejně jako jména funkcí či seznamů v nich jsou sázeny neproporcionálním písmem (`$acl_m0`).

Struktura dokumentu

Dokument obsahuje následující kapitoly:

Základy – obecný popis protokolu SMTP.
Metody filtrování – popis metod filtrování nevyžádané pošty v transakcích protokolu SMTP.
Další témata k zamyšlení – problémy, které souvisejí s problematikou filtrování.
Otázky a odpovědi – pokus o zapracování nejdůležitějších otázek a odpovědí.

Vzorová implementace metod filtrování s využitím softwaru Exim je uvedena v kapitole „Imple-mentace s využitím softwaru

Exim“.

Základy

V této kapitole naleznete informace o výhodách filtrování zpráv v průběhu příchozí transakce pro-tokolu SMTP. Tento způsob filtrování není až tak obvyklý, většinou se filtrování provádí až ve fázi směrování nebo doručení zprávy. Součástí této kapitoly je stručný popis transakce protokolu SMTP.

Proč filtrovat zprávy v průběhu transakce protokolu SMTP? Status Quo

Pokud dostáváte spam, zvedněte ruku a držte ji nahoře. Pokud dostáváte prostřednictvím elektronické pošty viry nebo jiný malware, také zvedněte ruku. Pokud dostáváte falešná oznámení
o doručení zprávy (Delivery Status Notification) typu „Zpráva nedoručitelná“, „Nalezen virus“ nebo „Prosíme potvrďte doručení“, která jsou reakcí na nikdy nezaslané zprávy, zvedněte ruku také. Tyto zprávy se obecně označují termínem zavlečený spam (collateral spam). Tato forma spamo-vých zpráv způsobuje při filtrování nejvíce problémů, protože se filtruje hůř než „standardní“ spam nebo malware a protože tyto zprávy mohou značně mást všechny příjemce, kteří nemají dosta-tečné schopnosti porozumět jednotlivým záhlavím zprávy. Zvýšený zájem uživatelů způsobují hlavně falešné zprávy o virech. Nicméně platí, že pokud uživatel dostává pravidelně větší množ-ství takových falešných zpráv, začne je postupem času ignorovat a to může vést k přehlédnutí anebo ignorování vážně míněných legitimních varování (například od správce sítě). A jako posled-ní tělocvičný prvek poprosíme ty z vás, kteří díky chybné klasifikaci spamovým filtrem nebo anti-virovým programem přišli o legitimní zprávu, zvedněte prosím do výšky vaši nohu.
Pokud jste před čtením této kapitoly stáli a pořád stojíte, je to možná tím, že nemáte úplně jasno v tom, co se může přihodit zprávám elektronické pošty. Pokud jste již prováděli nějaký způsob filtrování spamu, třeba jen ručním přesouváním zpráv do složky s odstraněnou poštou, nebo jste prováděli experimenty s primitivními filtrovacími postupy, jako jsou například seznamy zakáza-ných položek služby DNS (např. SpamHaus, SPEWS nebo SORBS), existuje jistá pravděpodobnost, že jste smazali i legitimní zprávy.

Příčina

Spam, stejně jako mnoho dalších artefaktů chtivosti, je sociální nemoc. Říkejme jí chřipka nebo jakkoliv jinak, základní princip je totiž podobný. Spočívá v tom, že nižší formy života hledají způ-sob, jak zničit větší ekosystém, a pokud se jim to podaří, dojde to do takového konce, kdy zničí i svoje vlastní útočiště.

Teď zanechme filozofování: Vy – správce poštovního systému – čelíte velmi konkrétnímu dilema
tu při hledání způsobů, jak se s tímto odpadem vypořádat. Při zpracování pošty existují v konvenčním způsobu, jakým jsou zprávy zpracovávány a předává-ny mezi jednotlivými komponentami softwaru pro doručování nebo směrování zpráv, jistá ome-zení. V obvyklé implementaci přijímá většinu nebo všechny příchozí zprávy, které jsou adresová-ny pro danou doménu, jeden nebo více poštovních serverů. Tyto servery často směrují zprávy pro další zpracování na jeden nebo více dalších serverů ve vnitřní síti nebo je přímo doručují do poš-tovních schránek uživatelů. Pokud kterýkoliv ze serverů zjistí, že není schopen provést požado-vané doručení zprávy nebo jinou funkci, vygeneruje a vrací zpět na adresu odesílatele původní zprávy oznámení o stavu doručení.

Současně s tím, jak organizace začaly nasazovat antivirové a antispamové programy, bylo obvyk-le cestou nejmenšího odporu zjištěno, že nejvhodnější umístění pro tyto programy se nachází přímo na cestě doručování zpráv například mezi jednotlivými poštovními servery nebo hostitel-skými systémy doručování zpráv anebo softwarem doručování zpráv. Obvyklým způsobem filtro-vání spamu je například *směrování* zpráv přes software SpamAssassin ještě před vlastním doruče-ním zprávy do poštovní schránky uživatele nebo celý systém filtrování spamu spoléhá na schop-nosti filtrování přímo v poštovních klientech jednotlivých uživatelů.

Možnosti, které se u těchto způsobů filtrování nabízejí po rozpoznání spamové nebo zavirované zprávy, jsou poměrně omezené:

Odesílateli je možné vrátit oznámení o stavu doručení. Problém je, že téměř veškerý spam i zavirované zprávy obsahují falešnou adresu odesílatele. Pokud na takové zprávy bude systém odpovídat, s největší pravděpodobností se to dotkne nevinných obětí – je to srov-natelné s varováním babičky ze Švédska, která používá počítač se systémem Mac OS X a neví nic o počítačích, že její počítač je infikován červem Blaster. Dále to pak znamená, že i vy se stanete generátorem zavlečeného spamu.
Zprávu je možné přesunout do spamového koše bez zaslání oznámení o stavu doručení zpět odesílateli. To může být problém v případě falešných pozitivních detekcí, protože ani odesílatel, ani příjemce se nikdy nedozví, co se se zprávou stalo (a příjemce se ani nedozví, že zpráva kdy existovala).
V závislosti na tom, jakým způsobem uživatelé přistupují ke svým zprávám (přistupují s využitím protokolu IMAP, využívají webový přístup, ale nikoliv v případě, kdy zprávy sta-hují s využitím protokolu POP3), je možné potenciální spam přesouvat do zvláštní složky, která může být například součástí nastavení poštovního účtu.

Poslední možnost je zřejmě nejlepší, ale v závislosti na tom, jak pravidelně nebo nepravidelně složku s potenciálním spamem uživatel prochází, může dojít k přehlédnutí nebo nechtěnému sma-zání legitimních zpráv, které byly chybně klasifikovány a přesunuty do této složky.

Řešení

Jak určitě sami tušíte, jediné řešení tohoto problému je provádět filtrování spamu a antivirovou kontrolu při přijímání zprávy poštovním serverem, tj. v průběhu probíhající transakce protokolu SMTP se vzdáleným hostitelem. Pokud se v průběhu transakce ukáže, že zpráva je nevyžádaná, je možné provést příkaz reject protokolu SMTP a dále není nutné řešit dilemata popsaná výše. Výsledkem je, že:

Doručení většiny nevyžádaných zpráv je možné zastavit již v průběhu transakce protoko-lu SMTP ještě před vlastním přijetím obsahu zprávy, což šetří šířku pásma i čas procesoru.
Při filtrování je možné využít například zpožďování transakcí protokolu SMTP nebo grey-listing, což jsou metody, které se nedají provádět v pozdějších fázích zpracování zpráv.
V případě nedoručení zprávy (například díky chybně zadané adrese příjemce zprávy) je

možné uvědomit odesílatele zprávy bez generování nevyžádaného zavlečeného spamu. V tomto dokumentu naleznete kromě jiného informace o tom, jak zabránit nepřímému generování zavlečeného spamu díky odmítnutí zpráv z důvěryhodných zdrojů, jako jsou například mailinglisty nebo servery a poštovní účty v jiných doménách.

Poznámka

Nedůvěryhodné hostitelské systémy mohou v případě odmítnutí zprávy generovat zavle-čený spam. Pokud takový hostitelský systém nepracuje v režimu open proxy nebo open relay, pravděpodobně doručuje zprávy pouze od legitimních odesílatelů, jejichž adresy jsou platné. Pokud systém pracuje v režimu open proxy nebo open relay, je lepší zprávu odmítnout a nechat ji shnít uloženou v odchozí frontě zpráv tohoto systému.

- Je umožněna ochrana před zavlečeným spamem z jiných adres (například falešných zpráv typu „Máte virus“, které zasílá antivirový program). Tak jo, už můžete dát ruce dolů. A pokud stojíte jen na jedné noze, už si můžete stoupnout na obě.

Pro a proti

Některé metody filtrování jsou pro použití v průběhu transakce protokolu SMTP vhodnější než

jiné. Některé jsou prostě lepší než jiné. Téměř všechny však mají své pro a proti.Tyto klady i zápory platí i pro metody popsané v tomto dokumentu. Platí například následujícítvrzení:

Je možné říci, že kontroly prováděné s využitím systému DNS mohou znevýhodnit jednot-livé odesílatele zpráv v závislosti na poskytovateli jejich internetového připojení a ne na základě obsahu příslušných zpráv.
Dále je možné říci, že metody zpožďování transakcí protokolu SMTP nebo greylisting jsou snadno překonatelné a s postupem doby budou méně účinné a budou degradovat kvalitu služby pro legitimní zprávy.
Dá se říci, že některá schémata autorizace odesílatele, jako například SPF (Sender Policy Framework) dávají poskytovatelům připojení k Internetu možnost „zablokovat“ si své zákazníky v tom smyslu, že neumožní dostatečně využívat poštovních služeb těm, kteří se

připojují v různých sítích, nebo těm, kteří přeposílají zprávy mezi různými hostitelskými systémy.

Od těchto protikladů se budeme raději držet dál a v dokumentu naleznete popis fungování růz-ných dostupných metod spolu s vysvětlením všech možných vedlejších efektů a s popisem auto-rových zkušeností s jejich využíváním.

V současnosti existují některé způsoby filtrování, které jsou v tomto dokumentu v tichosti pominuty:

- Systémy typu výzva/odezva (challenge/response), jako je např. TDMA. Tyto systémy nejsou pro filtrování protokolu SMTP s využitím časových zpoždění vhodné, protože tyto systémy pracují na principu přijetí zprávy a následném vracení požadavku o potvrzení na adresu odesílatele zprávy z příkazu MAIL FROM:. Tato metoda je již za hranicí rozsahu tohoto dokumentu.

  Poznámka

  Autor dokumentu se nedomnívá, že systémy typu výzva/odezva jsou obecně vhodné. Jednak generují zavlečený spam, vyžadují speciální pozornost u zpráv zasílaných auto-matizovanými zdroji, jako jsou například měsíční bankovní výpisy, a degradují použi-telnost elektronické pošty díky tomu, že odesílatelé legitimních zpráv se nechtějí zatě-žovat s požadavky na potvrzení nebo nevědí, jak na požadavek na potvrzení reagovat, a tudíž se legitimní zprávy, pro které uživatel z nějakého důvodu nepotvrdí požadavek na potvrzení, mohou ztratit.

Bayesovy filtry. Tyto filtry vyžadují před nasazením v produkčním prostředí trénink, který může být specifický pro konkrétního uživatele a konkrétní jazyk. Díky tomu tento způsob není vhodný pro použití v průběhu transakce protokolu SMTP.
Technologie mikroplateb, které nejsou pro blokování nevyžádané pošty vhodné, dokud nebudou virtuální *poštovní známku*

obsahovat úplně všechny odesílané zprávy. (Zatím to vypadá, že se využívají pro opačné účely – tzn. pro přijetí takové zprávy, která obsahuje známku a která by byla za jiných okolností odmítnuta.)

Jednotlivé metody uvedené v dokumentu jsou při detekci nevyžádané pošty co možná nejpřes-nější a jsou vybrány tak, aby v co nejmenší míře zabránily vzniku falešných pozitivních detekcí. Zprávy elektronické pošty jsou pro uživatele důležité a ti jim věnují svůj čas a úsilí. Metody a nástroje, které odmítají velké množství legitimních zpráv, ukazují na nedostatek respektu jak k lidem, kteří zprávy zasílají, tak k Internetu jako celku.

> Poznámka
>
> Názor autora je v přímém rozporu s názorem velkého počtu tzv. spamových hacktivistů, například se správci seznamu zakázaných položek SPEWS (viz *http://www.spews.org/*). Jed-ním z přímých cílů správců tohoto seznamu je bojovat proti zavlečenému spamu hlavně vyvíjením tlaku na poskytovatele připojení k Internetu.

Toto obvykle nejsou použitelné metody. Vezměme v úvahu rozvojové země a předpokládejme fakt, že v takových zemích jsou poskytovateli širokopásmového připojení regulované monopoly. Myslím, že tento příklad ilustruje přesnou příčinu problému s důvěryhodností poskytovatelů.
Obecně však platí, že existují lepší a přesnější způsoby, jak filtrovat nevyžádanou poštu.To platí zvláště pro filtrování s využitím časového zpoždění, protože koncoví uživatelé obvyklenemají možnost ovlivnit jednotlivá kritéria, která jsou pro filtrování zpráv použita.

Transakce protokolu SMTP

Protokol SMTP slouží pro doručování zpráv v prostředí Internetu. Podrobný popis protokolu naleznete v dokumentu RFC 2821 a také v dokumentu Davea Rockera o architektuře internetové pošty (*http://www.brandenburg.com/specifications/draft-crocker-mail-arch-00.htm*).

Doručení poštovní zprávy vyžaduje provedení transakce protokolu SMTP mezi hostitelem, který se připojuje (klient), a hostitelem, který připojení přijímá (server). Pro účely této diskuse budeme předpokládat, že hostitel, který se připojuje, je klientský počítač a hostitel, který připojení přijímá, je poštovní server.

V průběhu transakce protokolu SMTP klient zasílá příkazy protokolu SMTP jako například EHLO, MAIL FROM:, RCPT TO: nebo DATA. Server na každý příkaz odpovídá numerickým kódem slo-ženým ze tří číslic, který udává, jestli byl příkaz přijat (kódy 2xx), jestli byl příkaz předmětem dočasného selhání nebo omezení (kódy 4xx) nebo jestli příkaz selhal definitivně nebo trvale (kódy 5xx). Za kódem následuje textové vysvětlení. Úplný popis těchto kódů je součástí doku-mentu RFC 2821.

Transakce protokolu SMTP se ve vzorovém případě skládá z následujících kroků:

Transakce protokolu SMTP

Klient	Server
Iniciuje připojení k serveru pomocí protokolu TCP.	Zobrazí úvodní zprávu protokolu SMTP, tzn. text, který začíná kódem . Tento kód udává, že server je připraven přijímat příkazy protokolu SMTP (nebo ESMTP, což je nadmnožina proto-kolu SMTP): 220 *název_domény* ESTMP...
Identifikuje se příkazem Hello, bu (v současnosti se nebo , za kterým následuje úplný název verzi příkazu Hello (), server ví, že klient je schopen zpracovat víceřádkové odezvy, a proto klientovi zašle několik	Přijme identifikaci kódem . Pokud klient použil rozšířenou domény: EHLO *úplný_název_domény* nepoužívá) řádků textu, ve kterých uvede jednotlivé vlastnosti serveru: 250-*úplný_název_domény* Hello ... 250-SIZE 52428800 250-8BITMIME 250-PIPELINING 250-STARTTLS 250-AUTH 250 HELP Pokud je součástí odezvy vlastnost , znamená to, že klient může od tohoto okamžiku zasílat více příkazů najed-nou bez toho, aby postupně čekal na odezvu na každý příkaz.
Začne transakci identifikací odesílatele (Envelope Sender): MAIL FROM:<*adresa@odesílatele*>	Zašle kód , který udává, že server akceptuje odesílatele.
Identifikuje jednotlivé příjemce zprávy (Envelope Recipient) povoleno, je předmětem dočasného omezení nebo odmítnuto).	Zasílá příslušnou odezvu na každý příkaz (, nebo , postupně jednoho po druhém s využitím příkazu: v závislosti na tom, jestli je doručení příslušnému příjemci RCPT TO:<*adresa@příjemce*>
Zašle příkaz , kterým serveru oznámí připravenost zaslat text zprávy.	Zašle kód , který indikuje, že příkaz byl dočasně přijat.
Přenese text zprávy, která začíná jednotlivými řádky záhlaví (dle RFC 2822), např. From:, To:, Subject:, Date:, Message-ID:. Záhlaví a tělo zprávy jsou odděleny prázdným řádkem. Konec zprávy je	Zašle kód , kterým udává, že zpráva byla přijata.

identifikován znakem tečka („.“) na zvláštním řádku.

Pokud klient zasílá více zpráv, následuje další příkaz . Odpojí se. V opačném případě zašle příkaz nebo se přímo odpojí.

Metody filtrování

V této kapitole naleznete popis různých metod filtrování nevyžádané pošty v průběhu probíhají-cí transakce protokolu SMTP a také popis různých vedlejších efektů, které doprovázejí použití těchto metod filtrování.

Zpoždění transakcí protokolu SMTP

Jednou z nejúčinnějších metod blokování spamu je zavedení zpoždění v příchozích transakcích protokolu SMTP. To je primitivní forma tzv. teergrubingu, viz *http://www.iks-jena.de/ mitarb/lutz/usenet/teergrube.en.html.*

Většina spamu a téměř všechny zavirované zprávy jsou na poštovní server doručovány pomocí specializovaného klientského softwaru, který je optimalizovaný pro přenosy vysokého objemu zpráv v krátkém časovém okamžiku. Tento klientský software je obvykle označován termínem *rat-ware*.

Aby bylo možné výše uvedené zadání splnit, autoři ratwaru obvykle využívají pár „zkratek“, které se trochu odchylují od specifikace protokolu SMTP z dokumentu RFC 2821. Jednou z takových zkratek je, že ratware bývá velmi netrpělivý, zvláště při komunikaci s poštovními servery, které odpovídají pomalu. Ratware například zkouší provést zaslání příkazu HELO nebo EHLO ještě předtím, než server zobrazí úvodní zprávu protokolu SMTP, nebo se pokusí o zřetězení několika příkazů protokolu SMTP ještě předtím, než server zveřejní informace o svých vlastnostech s vyu-žitím klíčového slova PIPELINING.

Někteří agenti přenosu zpráv (například Exim) takováto porušení konvencí protokolu SMTP pova-žují za *chyby synchronizace* a okamžitě ukončují příchozí připojení. Pokud takového agenta pře-nosu zpráv využíváte, pravděpodobně v logovacích souborech uvidíte mnoho záznamů o takto ukončených připojeních. Frekvence výskytu takových „chyb“ se zvyšuje současně s tím, jestli jsou před zobrazením úvodní textové zprávy protokolu SMTP prováděny kontroly, které zabírají jistý čas (například kontrola s využitím seznamu povolených položek služby DNS), protože klienti s rat-warem prostě nemohou čekat na odezvu ze serveru tak dlouho. (Mají spoustu práce, doručují spam lidem.)

Proto je možné do transakce vnést dodatečné zpoždění. Například je možné čekat:

20 sekund před zobrazením úvodní textové zprávy protokolu SMTP.
20 sekund po úvodním příkazu Hello (EHLO nebo HELO).
20 sekund po příkazu MAIL FROM:.
20 sekund po každém příkazu RCPT TO:.

Kde se ale vzalo právě 20 sekund? Proč ne zrovna minuta? Nebo několik minut? Dokument RFC 2821 udává, že klient, který zasílá zprávu, by měl čekat na každou odezvu protokolu SMTP něko-lik minut. Problém je, že některé hostitelské systémy, které přijímají zprávy (například ty, které vyu-žívají Exim), mohou provádět pro každou příchozí poštovní transakci ověření adresy odesílatele. Pokud některý uživatel z vaší domény zašle zprávu takovému hostitelskému systému, systém nej-prve kontaktuje poštovní server (identifikovaný MX záznamem) vaší domény a pokusí se ve vaší doméně transakcí protokolu SMTP ověřit adresu odesílatele. Výchozí maximální časový interval takového ověření adresy odesílatele je 30 sekund. Pokud by časové zpoždění trvalo takto dlouho, ověření adresy odesílatele by mohlo selhat a díky tomu by doručení zpráv z vaší domény mohlo být odmítnuto (obvykle chybou typu dočasného selhání, při které se pokus o doručení zprávy opa-kuje po dobu cca 5 dnů předtím, než je původní zpráva definitivně vrácena odesílateli).
Jinými slovy, 20 sekund je dostatečně dlouho na to, aby nezpůsobilo problémy při normálním doručování legitimních zpráv.Pokud nechcete taková časová zpoždění používat v každé transakci protokolu SMTP (například máte velmi vytížený server), je možné tato zpoždění používat selektivně. V těchto případech sezpoždění uplatní například pouze tehdy, když:

Dojde k problému při zjišťování informací o hostitelském systému pomocí DNS.
Dojde k jakémukoliv problému v průběhu transakce protokolu SMTP.
Dojde k pokusu o transakci u poštovního serveru, který má mezi jednotlivými MX zázna-my nejnižší prioritu. Ratware se obvykle zaměřuje na takové hostitelské systémy, legitimní agenti přenosu zpráv obvykle využívají poštovní servery s nejvyšší prioritou.

Použití selektivního zpoždění transakcí může být dobrým výchozím bodem při implementaci dalších kontrol, které budou popsány v následujících kapitolách. Pravděpodobně určitě nebudete chtít odmítnout příchozí poštovní zprávy pouze na základě výsledků kontroly s využitím seznamu SPEWS, nicméně na druhou stranu tato kontrola může poskytnout dostatek informací pro použití zpoždění transakcí. Výsledkem pak je, že doručování legitimních zpráv není (kromě lehkého zpoždění) nijak ovlivněno.

Na druhou stranu, pokud zjistíte jasné příznaky spamu (například kontrolami prováděnými v rámci transakce protokolu SMTP) a váš server to umožní, můžete jej nakonfigurovat pro další dodateč-né zpoždění například o délce 15 minut a teprve potom

doručení zprávy definitivně odmítnout.

> Poznámka
>
> Při zpožďování příchozích transakcí protokolu SMTP je nutné mít na paměti, že čekání a zpoždění současně blokuje prostředky protokolu TCP a také jiné zdroje serveru (napří-klad paměť). Pokud je server zatížený, implementace zpoždění transakcí protokolu SMTP může zvýšit riziko v případě útoku typu odepření služby. Řešením je například okamžité ukončení příchozího připojení bezprostředně poté, co je nezvratně zjištěno, že odesílate-lem je klient s ratwarem.

Použití zpoždění nemá jinou výhodu než přibrzdění spamera v úsilí dosáhnout zprávami co možná nejvíce uživatelů ještě před provedením dalších následných kontrol (například s využitím seznamu povolených položek služby DNS).

Z vlastních zkušeností autora je možné říci, že zavedení selektivního zpoždění transakcí protoko-lu SMTP způsobí chyby synchronizace u cca 50 % všech odmítnutých příchozích pokusů o doru-čení zpráv. Proto je možné říci, že téměř 50 % příchozí nevyžádané pošty je možné zastavit s vyu-žitím zpoždění transakcí protokolu SMTP.

Kontrola s využitím systému DNS

Údaje o konkrétním hostitelském systému je možné získat ještě předtím, než se začnou provádět příkazy protokolu SMTP, pomocí systému DNS (Domain Name System). Pro zjištění, jestli daná adresa IP porušuje nebo splňuje některá vybraná kritéria, je možné využít některé seznamy zaká-zaných položek systému DNS a pro zjištění integrity hostitelského systému se dá využít vyhledá-ní dopředného a reverzního záznamu systému DNS.

Předmětem kontrol s využitím systému DNS mohou být různé datové položky, postupně zjištěné v průběhu transakce protokolu SMTP (například název hostitelského systému, uvedený klientem v příkazu Hello). Více podrobností o jednotlivých položkách naleznete v dalších kapitolách.
Na místě je ovšem varování. Kontroly s využitím systému DNS nemusí být spolehlivé (například v případě, kdy daný server systému DNS neodpovídá) a ne vždy jednoznačně identifikují spam. Pokud je váš poštovní server velmi zatížený, mohou kontroly prodloužit dobu zpracování zpráv. Na druhou stranu tyto kontroly mohou poskytnout užitečné informace pro účely záznamu informací do logovacích souborů.

Seznamy zakázaných položek systému DNS

Seznamy zakázaných položek systému DNS (dříve také označované termínem „Real-time Black-hole lists“ podle původního seznamu z adresy mail-abuse.org) jsou zřejmě nejpoužívanějším nástrojem pro blokování spamu v průběhu transakce protokolu SMTP. Server, který zprávu přijí-má, provede jednou nebo i několik vyhledávání reverzních záznamů systému DNS v různých seznamech zakázaných položek, např. „dnsbl.sorbs.net“, „opm.blitzed.org“ nebo „lists.dsbl.org“. Pokud je daný záznam v seznamu nalezen, obvykle to znamená odmítnutí doručení zprávy.

> Poznámka
>
> Podobné seznamy se používají i pro jiné účely. Seznam „bondedsender.org“ obsahuje povo-lené (=důvěryhodné) položky systému DNS. Vlastníci adres protokolu IP těchto položek slo-žili finanční zálohu, která bude snížena v případě, kdy z dané adresy začnou odcházet spa-mové zprávy. Ostatní seznamy mohou obsahovat adresy protokolu IP pro konkrétní zemi, konkrétní poskytovatele připojení atd.

Kromě adresy DNS (záznam typu „A“) je možné vyhledat i záznam typu „TXT“, který obsahuje jed-nořádkový popis zakázané položky, kterou je vhodné zahrnout do odezvy odmítající doručení zprávy. Pokud to chcete vyzkoušet, je možné použít příkaz host, který je dostupný u většiny systé-mů Linux nebo UNIX:

```
host -t txt 2.0.0.127.dnsbl.sorbs.net
```

Podobných seznamů existují stovky, každý z nich má jiná kritéria pro zařazení položky do sezna-mu a jiné zásady pro vyřazení položky ze seznamu. Některé seznamy dokonce kombinují více kri-térií a při reverzním vyhledávání vracejí různá data v závislosti na konkrétním kritériu, které je použito u vyhledávané adresy. Reverzní vyhledání pro doménu *sbl-xbl.spamhaus.org* vrací adre-su 127.0.0.2 pro všechny adresy protokolu IP, které podle správců seznamu SpamHaus přímo patří spamerům nebo jejich poskytovatelům, vrací adresu 127.0.0.4 pro všechny hostitelské systémy, které fungují jako zombie, a vrací adresu 127.0.0.6 pro všechny servery, které pracují v režimu open proxy.

Některé seznamy bohužel obsahují rozsáhlé bloky adres protokolu IP, které nejsou přímo zodpovědné za generování spamu, nemají jasné zásady pro zařazení nebo vyřazení položek nebo obsahují chybné informace o zařazených adresách.

> Poznámka
>
> Například servery odchozí pošty největšího poskytovatele připojení k Internetu na světě, firmy Comcast, v době psaní tohoto dokumentu využívaly seznam SPEWS Level 1. Je urči-tě jasné, že Comcast chce ve své síti efektivně prosadit vlastní zásady přijatelného využí-vání (AUP – Acceptable use policy), nicméně tento seznam má dopad na 30 % všech

uži-vatelů Internetu v USA, což jsou z valné většiny „nevinní“ zákazníci, mezi nimiž je i autor tohoto dokumentu.

Co je však ještě horší, informace uveřejněné v části FAQ na stránkách seznamu SPEWS uvádějí: Většina seznamů z úrovně Level 1 obsahuje bloky adres, které přímo vlastní spa-meři nebo které podporují spamové operace s pouze několika nebo žádnými jinými zjištěnými legitimními zákazníky. Tato informace je technicky přesná, pokud předpokládáme,
že Comcast podporuje spamové operace, a pokud se zaměříme na slovo „žádnými“. Ve skutečnosti je tato informace naprosto zavádějící.

Slepá důvěra v takové seznamy může způsobit problémy při doručování pošty. Mnoho správců poštovních systémů místo odmítnutí doručení zprávy na základě jediné pozitivní odezvy od některého seznamu zakázaných položek systému DNS využívá seznamy daleko rafi-novaněji. Kontrola se provádí u více různých seznamů a jednotlivé pozitivní výsledky jsou ohod-noceny nějakou váhou. Pokud celkový součet vah pro danou adresu protokolu IP přesáhne jistou
prahovou hodnotu, doručování zpráv z této adresy je odmítnuto. Takovým způsobem využívá seznamy zakázaných položek systému DNS například filtrující software SpamAssassin. Kontrolu s využitím seznamů je také možné využít jako podmínku pro aktivaci zpoždění transak
ce protokolu SMTP. Pokud je daný hostitelský systém uveden v seznamu, server může zpožďovat odezvy na každý příkaz protokolu SMTP například o 20 sekund. Jako podmínku pro aktivaci zpož-dění je samozřejmě možné využít i jiná kritéria (viz kapitola „Zpoždění transakcí protokolu SMTP“).

Kontrola integrity záznamů systému DNS

Systém DNS je možné také využít tak, že se nejprve provede reverzní vyhledání na základě adresy protokolu IP systému, který navazuje spojení, a výsledný vyhledaný název se použije při následném dopředném vyhledání. Pokud je původní adresa protokolu IP obsažena ve výsledku dopředného vyhledání, je tím pro danou adresu protokolu IP ověřena integrita záznamů systému DNS. V opač-ném případě nejsou informace v systému DNS pro systém, který navazuje spojení, korektní.

Odmítnutí zpráv na základě kontroly integrity může být vhodnou volbou pro militantní členy poli-cie systému DNS, kteří si nastavují záznamy typu MX pro své vlastní osobní domény a neštítí se odmítat doručení legitimních zpráv jako součást taktiky, kterou chtějí donutit odmítnuté odesíla-tele ke kontrole a případné opravě jejich záznamů v systému DNS. Pro všechny ostatní může být kontrola integrity záznamů systému DNS použita jako jedna z více různých kontrol. Podobně jako v předchozí kapitole i u kontroly integrity platí, že to může být vhodná podmínka pro aktivaci zpoždění transakcí protokolu SMTP.

Kontroly transakce protokolu SMTP

V průběhu transakce protokolu SMTP je možné provádět různé typy kontrol příkazů i jejich argu-mentů, které vzdálený hostitel zasílá. Jednoduchým příkladem může být kontrola názvu hostitel-ského systému z části Hello.

I v případě, kdy dojde k rozhodnutí odmítnout pokus o doručení zprávy již v úvodních fázích transakce protokolu SMTP, není nutné provést odmítnutí hned. Místo toho je vhodné zpozdit ode-sílatele aktivací zpoždění až do příkazu RCPT TO: a zprávu odmítnout po tomto příkazu.

Důvodem je, že některé typy ratwaru neumí odmítnutí v úvodních fázích transakce protokolu SMTP správně vyhodnotit a snaží se zprávu doručit znovu. Většina ratwaru však opakované poku-sy vzdá v případě, kdy dojde k odmítnutí transakce po příkazu RCPT TO:.

Kontroly úvodní fáze Hello (HELO/EHLO)

Podle dokumentu RFC 2821 by měl být prvním příkazem transakce protokolu SMTP, který klient zasílá, příkaz EHLO (pokud není podporovaný, tak HELO), následovaný úplným názvem domé-ny odesílatele. Tomuto příkazu se také většinou říká pozdrav Hello. Pokud není k dispozici úplný
název domény odesílatele, může klient použít adresu protokolu IP uzavřenou v hranatých závor

kách takto: [1.2.3.4]. Tento způsob se nazývá „písmenná“ notace adresy IPv4.Jak je zřejmé, ratware v příkazu Hello použije úplný název domény pouze výjimečně. Ratware seobvykle spíše snaží skrýt identitu odesílajícího hostitelského systému nebo vygenerovat falešnéstopy v záhlaví „Received“ zprávy. Mohou to být například:

Neúplné názvy (názvy bez teček), například název domény příjemce bez teček.
Libovolné adresy IP (tzn. nekorektně uvedená adresa podle definice písmenné notace). Může to být například adresa hostitelského systému příjemce nebo prostě jakákoliv náhodná adresa.
Název domény příjemce zprávy nebo úplný název domény vašeho poštovního serveru.
Názvy jiných domén, například yahoo.com nebo hotmail.com.
Názvy neexistujících domén nebo názvy domén bez existujících poštovních serverů.
Prázdný název.

Jednoduché kontroly syntaxe HELO/EHLO

Některá narušení dokumentu RFC 2821 se zkontrolují snadno a jsou jasným důkazem, že odesíla-jící hostitelský systém využívá nějaký typ ratwaru. Transakci protokolu SMTP je tedy v takových případech možné odmítnout a ukončit buď okamžitě nebo až po příkazu RCPT TO:.

Nijak se neomezujte odmítat ty transakce, u kterých je v příkazu Hello použita adresa protokolu IP neuzavřená v hranatých závorkách. I když při kontrolách povolíte všechna možná doporučení a návrhy obsažené v dokumentu RFC 2821, určitě zaznamenáte, že adresa protokolu IP by měla být vždy uzavřena v hranatých závorkách.

> Poznámka
>
> Tato kontrola je za normálních okolností při boji se spamem velmi účinná, nicméně exis tují zprávy o tom, že některé instalace softwaru listserv firmy L-Soft používají v příkazu Hello adresu protokolu IP neuzavřenou v hranatých závorkách.

Hostitelským systémům, které v příkazu Hello použijí adresu protokolu IP nebo hostitelský název *vašeho* systému, je možné vrátit peprnou zprávu s odmítnutím. Důvod je jasný – jde o podvrh. V tomto případě je také možné odesílatele ztrestat extrémně dlouhým zpožděním transakce pro-tokolu SMTP, mohou to být i řádově hodiny.

Zkušenost autora říká, že *žádné* legitimní systémy v prostředí Internetu se neprezentují jiným systémům s využitím písmenné notace adresy protokolu IP (tzn. [x.y.z.w]). Platí, že všechny hos-titelské systémy, které přímo zasílají zprávy, by měly používat úplný název domény. Jediný pří-pad použití písmenné notace adresy IP autor zaznamenal při použití klientských poštovních pro-gramů (např. Ximian Evolution), které využívaly poštovní server jako odchozí server protokolu SMTP. Je samozřejmostí, že písmenné notace adresy IP jsou v takovém případě povoleny pouze pro adresy z místní sítě.

U neúplných názvů domény (názvy bez teček) je možné odmítnutí provést nebo i neprovést. Exi stují řídké výjimky, u kterých je i neúplný název domény legitimní. Stejně tak je možné transakci odmítnout v případě, kdy název hostitelského systému obsahuje neplatné znaky. V prostředí internetových domén jsou platnými znaky všechny alfanumerické znaky a pomlčka s tím, že pomlčka nesmí být použita jako první znak. (Stejně tak je možné za platný znak považovat podtržítko, protože se často využívá u chybně nakonfigurovaných klientů se systémem Windows.)

Pokud transakce začne přímo příkazem MAIL FROM: bez úvodního příkazu Hello, je řešení nasnadě: Slušní lidé nejprve pozdraví.Na poštovních serverech autora jsou odmítnuty všechny transakce, které nevyhoví kontrolám syn-taxe příkazu Hello. K odmítnutí transakce pak dojde po příkazu RCPT TO:. Mezitím následujízpoždění transakce o délce 20 sekund pro všechny ostatní příkazy protokolu SMTP (HELO/EHLO,MAIL FROM: a RCPT TO:).

Kontrola příkazu Hello s využitím DNS

Hostitelské systémy, které prošly prvními kontrolami příkazu Hello, je dále možné kontrolovat s využitím systému DNS. V této fázi je možné:

Provést dopředné vyhledání zadaného názvu a porovnat výsledek s adresou IP hostitelského systému, který navázal spojení.
Provést reverzní vyhledání adresy IP hostitelského systému, který navázal spojení a porov-nat výsledný název s tím, který byl uveden v příkazu Hello.

Pokud obě kontroly proběhnou bez problému, název byl úspěšně ověřen. Váš agent přenosu zpráv může mít implementovanou možnost provedení této kontroly. Software Exim (viz kapitolu „Implementace s využitím softwaru Exim“) umožní kontrolu provést nastave ním parametru `helo_try_verify_hosts = *` a vytvořit seznam řízení přístupu, který na základě podmínky `verify = helo` provede příslušnou akci. Tato kontrola je z pohledu výpočetního a síťového výkonu poněkud náročnější než prostá kon-trola syntaxe. Na rozdíl od kontroly syntaxe v tomto případě překlepy nebo chyby v názvech pokaždé nemusejí ukazovat na ratware, některé velké domény, například hotmail.com, yahoo.com nebo amazon.com velmi často používají příkazy Hello, které nejsou verifikovatelné.

Na serverech autora se provádí ověřování pomocí DNS v případě, kdy pro odesílatele ještě neby-lo (v závislosti na předchozích kontrolách) aplikováno zpoždění transakce. Pokud kontrola s vyu-žitím systému DNS selže, aplikuje se zpoždění 20 sekund pro každý další příkaz transakce proto-kolu SMTP. Současně je vytvořeno záhlaví s varováním „X-HELO-Warning“, které je později při-dáno do zprávy a zvyšuje u softwaru SpamAssassin váhu pro možné odmítnutí zprávy po přijetí vlastních dat zprávy.

Kontroly adresy odesílatele

Poté, co klient uvede v příkazu MAIL FROM: <adresa_odesílatele> adresu odesílatele, je možné provést ověření platnosti této adresy několika způsoby.

> Poznámka
>
> Speciální případ je prázdná adresa odesílatele, tzn. MAIL FROM: <>, která se používá u oznámení o stavu doručení nebo jiných automaticky generovaných odezev. Prázdná adresa by měla být při kontrolách vždy akceptována.

Kontrola syntaxe adresy odesílatele

Vyhovuje zadaná adresa formátu *<uživatel@doména>*? Je část *doména* částí syntakticky správně zapsaného úplného názvu domény? Většina agentů přenosu zpráv – MTA – tyto kontroly prová-dí automaticky.

Kontrola falešné adresy

Pokud uživatelé zasílají veškerou odchozí poštu pomocí několika málo poštovních serverů, je možné odmítnout zprávy z jiných hostitelských systémů, kde je název domény odesílatele shod-ný s vaší doménou.

Obecnější alternativa této kontroly je použití SPF (Sender Policy Framework).

Kontrola platnosti adresy odesílatele

Pokud je adresa místní (=z místní domény), je část adresy před znakem @ shodná s existující poš-tovní schránkou v místní doméně?Pokud je adresa cizí, existuje doména, uvedená za znakem @?

Kontrola platnosti odesílatele zpětným voláním (Sender Callout Verification)

Tento mechanismus implementují někteří agenti přenosu zpráv, například Exim nebo Postfix. Mechanismus ověřuje část adresy před znakem @. Terminologií použitou v softwaru Postfix se tato kontrola nazývá „Ověření adresy odesílatele“ („Sender Address Verification“).

Při této kontrole poštovní server, který přijímá zprávu, kontaktuje s využitím záznamu typu MX poštovní server v doméně získané z adresy odesílatele a pokusí se provést druhou transakci pro-tokolu SMTP, jako by chtěl doručit do vzdáleného systému zprávu na tuto adresu odesílatele. Žád-nou zprávu samozřejmě nezasílá a po přijetí nebo odmítnutí příkazu RCPT TO: transakci ukončí.

Pro takové ověření adresy odesílatele ve vzdáleném systému Exim používá prázdnou adresu odesílatele. Cílem je zjistit, jestli by vzdálený systém akceptoval případné oznámení o stavu doručení zaslané na adresu odesílatele.

Software Postfix při ověření adresy odesílatele využívá jako adresu místního odesílatele adresu *<postmaster@doména>*, kde část *doména* je převzata ze systémové proměnné *$myorigin*. Z toho důvodu je vhodné s takovou adresou odesílatele zacházet stejným způsobem jako s prázdnou adresou (tzn. nepoužít zpoždění transakce protokolu SMTP, nepoužít greylisting, ale vyžadovat signaturu odesílatele v adrese příjemce). Podrobnosti o této implementaci naleznete v příloze A.

Tato kontrola pro jednoznačné rozhodnutí o přijetí nebo odmítnutí příchozí zprávy rozhodně nestačí. Neplatné adresy odesílatele obsahují v některých případech i legitimní zprávy (například automaticky generované výpisy z účtu). Nešťastným vedlejším efektem spamu je i to, že někteří uživatelé mají snahu komolit adresu odesílatele ve své odchozí poště (to však častěji ovlivní záhla-ví „From:“ zprávy, nikoliv skutečnou adresu odesílatele v příkazu MAIL FROM:).

Je nutné mít na paměti, že tato kontrola pouze ověří platnost adresy, nikoliv autenticitu odesíla
tele zprávy. Jako poslední poznámku je nutné uvést, že existují systémy, např. „aol.com“, které zařazují do seznamu zakázaných položek všechny vzdálené systémy, které se snaží provést ověření adresy odesílatele. Takové systémy jsou často obětí spamu, který obsahuje podvrženou, avšak platnou adresu odesílatele, a výsledkem je velké množství požadavků na ověření platnosti odesílatele. Takovým způsobem je možné provádět i distribuované útoky odepření služby a vy se sami snad-no takovým způsobem můžete stát loutkou v rukou spamerů.

Kontroly adresy příjemce

Určitě si řeknete, že taková kontrola je jednoduchá. Adresa příjemce je buď platná, a to znamená doručení zprávy, nebo neplatná a v takovém případě dojde k jejímu odmítnutí. Zkusíme si však vše rozebrat podrobně.

Obrana proti open relay

Poštovní server by nikdy neměl doručovat zprávy ze vzdálených systémů do jiných vzdálených systémů! (Pokud ovšem není odesílatel dostatečným způsobem autorizován.)
To zní jako samozřejmost, ale často se tato vlastnost přehlíží. Ne každý má také dokonalé znalosti všech internetových standardů, které mají vztah k adresám elektronické pošty nebo způsobům doručování.

Pokud si nejste jisti, jestli váš agent přenosu zpráv náhodou nefunguje v režimu open relay, může-te jej zkusit otestovat s využitím domény „relay-test.mail-abuse.org“. V příkazovém řádku napište příkaz:

```
telnet relay-test.mail-abuse.org
```

To je služba, která s využitím různých testů ověří, jestli je váš SMTP server schopen přesměrová
vat zprávy na vzdálené poštovní adresy, a vyzkouší i různé jiné způsoby přesměrování. Zabránit, aby servery nepracovaly v režimu open relay, je extrémně důležité. Pokud server pra-cuje jako open relay a spameři jej objeví, zcela určitě se vzápětí dostanete na různé seznamy zaká-zaných položek systému DNS. Pokud se o vás správci takových seznamů dozví (buď si server sami otestují nebo na základě upozornění), určitě v seznamech budete uvedeni na velmi dlouhou dobu.

Kontrola adresy příjemce

Tohle zní jednoduše, ale nemusí tomu vždy tak být.

Pokud jsou poštovní účty a poštovní schránky uloženy na jediném příchozím poštovním serveru,

stačí ověřit část adresy před znakem @ a nalézt, jestli taková poštovní schránka existuje. Žádný problém.
Existují dvě situace, kdy je ověření adresy příjemce obtížnější:

Váš server slouží jako záložní poštovní server pro doménu příjemce.
Váš server přesměrovává veškerou poštu pro vaši doménu na jiný server (předpokládejme

na server ve vnitřní síti). Alternativou ke kontrole adresy příjemce je příjem všech adres v rámci dané domény, což ovšem znamená, že cílový server může generovat oznámení o stavu doručení pro neplatné adresy. To dále znamená, že se stanete generátorem zavlečeného spamu.
Teď se zaměříme na to, jak ověřit adresy příjemce u výše zmíněných situací.

Ověření příjemce zpětným voláním

To je mechanismus, který používají někteří agenti přenosu zpráv, například Exim nebo Postfix pro ověření části adresy příjemce před znakem @ (princip fungování naleznete v části Kontrola plat-nosti odesílatele zpětným voláním). V terminologii softwaru Postfix se tato kontrola nazývá „Ově-ření adresy příjemce“ („Recipient Address Verification“).

V tomto případě se server pokusí kontaktovat cílový hostitelský systém a ověřit všechny adresy příjemců ještě předtím, než server přijme příkaz RCPT TO: od hostitelského systému, který zprá-vu zasílá.

Takové řešení je jednoduché a elegantní. Funguje u každého agenta přenosu zpráv, který je spuš-těn na cílovém hostitelském systému bez nutnosti přístupu do adresářových služeb. Pokud navíc takový agent přenosu zpráv provádí kontrolu adresy příjemce s využitím fuzzy logiky (např. ser-very Lotus Domino), kontrola přesně zjistí, jestli je adresu příjemce možné akceptovat nebo ne, což ne vždy může platit pro ostatní metody popsané dále.

Ujistěte se, že kontrola nechává netknutou původní adresu odesílatele pro případnou kontrolu zpětným voláním. V opačném případě nemusí být odezva od cílového hostitelského systému přes-ná. V takových případech může taková kontrola například odmítnout vracené zprávy (zprávy bez odesílatele) pro systémové uživatele nebo aliasy.
Tento mechanismus podporují kromě jiných agentů přenosu zpráv i Exim a Postfix.

Adresářové služby

Dobrým řešením je také využití adresářových služeb (tj. jednoho nebo více serverů LDAP), které může agent přenosu zpráv dotazovat. Většina nepoužívanějších agentů přenosu zpráv podporuje protokoly LDAP, NIS nebo různé jiné backendové systémy, které je možné použít pro poskyto-vání informací o uživatelských účtech.

Hlavním problémem při použití adresářových služeb je, že cílový hostitelský systém musí adresářovou službu využívat pro mapování uživatelských jmen na poštovní schránky, a pokud cílový systém adresářové služby nevyužívá, je nutné provést příslušnou konfiguraci.

Replikované seznamy poštovních schránek

Pokud pro vás ani jedna možnost není použitelná, je nutné se vrátit k „adresářové službě chudé-ho správce“, což je periodické kopírování aktuálního seznamu poštovních schránek ze systému, kde jsou uloženy, na poštovní server. Agent přenosu zpráv pak tento seznam využije při ověřo-vání platnosti adres příjemců v příkazu RCPT TO:.

Pokud hostitelský systém, na kterém jsou uloženy poštovní schránky, využívá nějakou verzi ope-račního systému UNIX nebo Linux, je možné vytvořit skript, který seznam poštovních schránek vygeneruje, například z lokálního souboru /etc/passwd, a pak jej překopíruje na poštovní server s využitím příkazu scp z balíku nástrojů OpenSSH. Následně je možné nechat takový skript auto-maticky spouštět v zadaných časových intervalech pomocí nástroje cron.

Obrana proti slovníkovému útoku

Slovníkový útok je termín, který popisuje takové transakce protokolu SMTP, při kterých se odesí-lající hostitelský systém snaží pomocí řady příkazů RCPT TO: zjistit možné adresy příjemců na základě obecných jmen (obvykle podle abecedy, začíná se jménem „aaron“, někdy slovník obsa-huje náhodná jména nebo se začíná někde uprostřed abecedy). Pokud danou adresu server při-jme, spamer si ji přidá do svého seznamu.

Některé poštovní servery, obzvláště velkých organizací, jsou častými oběťmi takových útoků. Z pohledu spamera je pravděpodobnost uhádnutí nějakého uživatelského jména daleko větší u velké společnosti než u malé firmy s několika uživateli.

Efektivní způsob, jak takovým útokům čelit, je používat zvětšující se zpoždění transakce po každém neúspěšném pokusu o uhádnutí adresy. První neexistující příjemce například způsobí zpoždění 20 sekund, další způsobí zpoždění 30 sekund atd.

Akceptace jediného příjemce pro oznámení o stavu doručení

Legitimní oznámení o stavu doručení by mělo být zasíláno pouze na adresu jediného příjemce – původce původní zprávy, která způsobila vygenerování oznámení. Připojení je možné ukončit v případě, kdy je adresa odesílatele prázdná a existuje více

příjemců.

Greylisting

Základní princip greylistingu je uveden v dokumentu Evana Harrise na adrese *http://projects. puremagic.com/greylisting/.*

Jak to funguje

Podobně jako u zpoždění transakcí protokolu SMTP je i greylisting jednoduchý, ale vysoce efek-tivní mechanismus pro odfiltrování zpráv, které jsou zasílány ratwarem. Základní myšlenkou grey-listingu je kontrola, jestli mezi odesílatelem a příjemcem zprávy existuje nějaký vztah (bráno čistě
z pohledu zasílání zpráv). Pro většinu legitimních zpráv takový vztah existuje a jejich doručení

proto probíhá normálně.Pokud zatím takový vztah neexistuje, doručení zprávy je dočasně odmítnuto (odezvou protokoluSMTP číslo 451). Legitimní agenti přenosu zpráv se na tuto odezvu zachovají v souladu s logikoudoručování zpráv, tzn. za nějaký časový interval se o doručení pokusí znovu.

> Poznámka
>
> Je to sice výjimečné, ale i některé „legitimní“ hromadné odesílače zpráv (např. groups.yahoo.com) dočasně odmítnutý přenos neopakují. Evan Harris sestavil seznam tako-vých serverů, který je vhodné použít pro seznam povolených položek. Seznam naleznete zde: *http://cvs.puremagic.com/viewcvs/greylisting/schema/whitelist_ip.txt?view=markup.*

Ratware se na rozdíl od legitimního poštovního serveru pokusí o opakovaný přenos okamžitě

nebo to vzdá a zkouší další oběť ze svého seznamu adres.Pro jednoznačnou identifikaci vztahu mezi odesílatelem a příjemcem zprávy se sbírají tři základníinformace, které se souhrnně označují jako *triplet*. Je to:

Odesílatel zprávy z příkazu MAIL FROM:.
Adresa protokolu IP hostitelského systému odesílatele.
Příjemce zprávy z příkazu RCPT TO:.

Při pokusu o doručení zprávy, který je dočasně odmítnut, se triplet uloží do vyrovnávací paměti. Zde zůstane dočasně uložen po daný časový interval (obvykle 1 hodina). Po této době je přene-sen do seznamu povolených položek a další pokus o doručení již proběhne úspěšně. Pokud se během dalšího časového intervalu (například 4 hodiny) nevyskytne další pokus o doručení, triplet je z vyrovnávací paměti odstraněn.

Pokud se triplet, který byl součástí seznamu povolených položek, nevyskytne při doručování za dlouhý časový interval (minimálně 1 měsíc, to stačí na pokrytí například měsíčních výpisů z účtů atd.), je definitivně odstraněn, aby seznam tripletů nerostl donekonečna.

Hodnoty časových intervalů jsou převzaty přímo z původního dokumentu Evana Harrise. Praxí však bylo zjištěno, že časový interval pro dočasné uložení tripletu v seznamu povolených polo-žek je vhodné zvětšit, protože někteří poskytovatelé (například *earthlink.net*) opakují pokus
o doručení zprávy třeba až za 6 hodin.

> Poznámka
>
> Vetší organizace obvykle pro odchozí poštu využívají více poštovních serverů. Pro okamžité doručení může být například použita celá skupina serverů. Pokud první pokus
> o doručení selže, zpráva je předána na záložní server, který využívá delší fronty zpráv. U takových organizací selžou první dva pokusy o doručení dané zprávy.

Greylisting pro více poštovních serverů

Pokud organizace provozuje více poštovních serverů a každý server spravuje svou databázi tripletů, pak platí:

- První doručení zprávy od daného odesílatele jednomu z uživatelů poštovního serveru může být teoreticky zpožděno maximálně *N* krát původní zpoždění (1 hodina), kde *N* je počet poštovních serverů. Je to díky tomu, že pokus o další doručení zprávy může být opa-kován s využitím jiného poštovního serveru, než je ten, který vrátil kód 451 na první pokus o doručení zprávy. Hostitelský systém odesílatele může v nejhorším případě opakovat pokusy o doručení zprávy na první poštovní server za delší dobu, než jsou 4 hodiny, nebo až po vypršení platnosti tripletu, což způsobí, že pokus o doručení je opakovaně odmítán,
  dokud to hostitelský systém odesílatele nevzdá (obvykle po 4 dnech).

  V praxi je to velmi nepravděpodobné. Pokud je pokus o doručení dočasně odmítnut, ode sílatel se bezodkladně snaží o doručení s využitím jiného poštovního serveru ze seznamu záznamů typu MX pro danou doménu. Díky tomu může zprávu po úvodní hodině přijmout

kterýkoliv jiný poštovní server domény.

- V případě, že triplet byl přesunut do seznamu povolených položek na jednom poštovním serveru, další zpráva se stejným tripletem podstoupí stejný cyklus dočasného odmítnutí a zpoždění v případě, že je doručována pomocí jiného poštovního serveru.

Z těchto důvodů je vhodné databázi tripletů implementovat jako sdílenou pro všechny poštovní servery. Pokud však na systému s databází dojde k nějakému výpadku, je nutné zajistit odpoví-dající akci (například povolení všech pokusů o doručení). Také je možné využít nějaký způsob replikace a nechat server SMTP použít při vyhledávání v databázi tripletů jiný replikovaný server.

Výsledky

Z vlastní zkušenosti autora greylisting zvládne odfiltrovat cca 90 % pokusů o doručení nevyžáda-né pošty i *poté*, co byly aplikovány dříve popsané kontroly transakcí protokolu SMTP. Pokud se greylisting použije jako technologie první obrany, pravděpodobně zachytí ještě vyšší procento nevyžádané pošty.

Při použití greylistingu téměř nedochází ke vzniku falešných pozitivních detekcí. Všichni hlavní agenti přenosu zpráv provádějí po prvním dočasném odmítnutí zprávy opakování doručení, což je chování, které v důsledku vede k úspěšnému doručení legitimních zpráv.

Nevýhodou greylistingu je, že legitimní zpráva od odesílatele, který danému příjemci ještě žádnou zprávu v minulosti nezaslal, bude doručena s cca hodinovým zpožděním (nebo s větším zpožděním při použití více poštovních serverů).

Schémata autorizace odesílatele

Pro ověření odesílatele byla vytvořena různá schémata, která ověřují nejen platnost adresy odesílatele, ale také autenticitu odesílatele. Vlastník internetové domény specifikuje určitá kritéria, která musí být splněna, aby bylo zaručeno bezpečné doručování v rámci odesílatelů v dané doméně.

Dvě uveřejněná schémata zahrnují:

MX záznamy typu MAIL-FROM, vytvořené Paulem Vixiem <*paul@vix.com*>
Záznamy typu RMX (Reverse Mail Exchange), které jsou doplněním systému DNS. Auto-rem je Hadmut Danisch <*hadmut@danish.de*>

U obou schémat musí všechny zprávy od uživatele *uživatel@doména.com* pocházet z hostitel-ských systémů v zóně DNS domény <doména.com>.Obě schémata se dále vyvíjela, ale kromě jiného také větvila.

Technologie Sender Policy Framework

SPF (předtím se využíval název „Sender permitted From“) je pravděpodobně nejznámější schéma autorizace odesílatele. Je založeno na principech původních schémat, popsaných výše, ale umož-ňuje větší flexibilitu v kritériích, která může vytvářet a zveřejňovat vlastník domény.

Informace SPF je zveřejněna jako záznam typu TXT v zóně DNS nejvyšší úrovně pro danou doménu. Záznam může obsahovat následující informace:
Hostitelské systémy, ze kterých je povoleno zasílat zprávy.
Přítomnost signatury GPG (GNU Privacy Guard) v odchozích zprávách z domény.
Jiná kritéria, podrobnosti viz *http://spf.pobox.com/*.

Struktura záznamu typu TXT prochází neustálým vývojem, nicméně základní vlastnosti jsou již sta-noveny. Záznam začíná řetězcem v=spfl, za kterým následují modifikátory:

a – adresa protokolu IP vlastní domény je platným odesílatelem
mx – server příchozí pošty pro doménu je také platným odesílatelem
ptr – pokud reverzní dotaz s adresou protokolu IP vede na název v rámci doménové části adresy odesílatele, je to platný odesílatel

Každý modifikátor může mít prefix plus (+), minus (-), otazník (?) nebo tilda (~), který udává auto
ritativní zdroj, neautoritativní zdroj, neutrální zdroj a pravděpodobně neautoritativní zdroj. Každý modifikátor může být rozšířen dvojtečkou, za kterou následuje alternativní název domény. Pokud například využíváte doménu Comcast, může zóna DNS obsahovat například následující řetězec: -ptr:client.comcast.net ptr:comcast.net, ten udává, že odchozí zpráva nikdy nebude pocházet z hostitelského systému, který má adresu cokoliv.client.comcast.net, ale může pocházet z hostitelských systémů s adresami cokoliv.comcast.net.

Informace SPF je aktuálně publikována pro většinu nejznámějších internetových domén, jako jsou například aol.com, altavista.com, dyndns.org, earthlink.net nebo google.com.Schémata autorizace odesílatele (obecně) a technologie SPF (částečně) nejsou přijímány univer-zálně. Námitkou proti používání těchto technologií je, že vlastníci domény

mohou takovým způ-sobem vytvořit monopol na odesílání zpráv od svých uživatelů a zákazníků.

Další nevýhodou je, že technologie SPF koliduje s tradičním přeposíláním zpráv – hostitelskýsystém, který zprávu přeposílá, nemusí mít na základě informací ze SPF v doméně odesílateleoprávnění takovou akci provést. Tento problém částečně řeší technologie SRS (Sender RewritingScheme), při jejímž použití systém, který zprávu přesměrovává, současně upravuje adresu odesí-latele do formátu:

uživatel=zdrojová.doména@přeposílaná.doména

Schéma Sender-ID firmy Microsoft

Technologie se podobá technologii SPF v tom, že kritéria pro akceptaci jsou uložena v záznamu typu TXT v zóně DNS domény odesílatele. Místo využití jednoduchých klíčových slov jsou infor-mace MS CIDE uloženy v rozsáhlých strukturách ve formátu XML. Schéma XML je zveřejněno pod licencí firmy Microsoft.

Zatímco technologie SPF je primárně používaná pro kontrolu adresy odesílatele z části MAIL FROM: dané zprávy, MS CIDE se primárně využije pro ověření záhlaví vlastní zprávy dle definice v dokumentu RFC 2822. Takovou kontrolu je však možné nejdříve provést až po doručení vlast-ních dat zprávy před zasláním poslední odezvy s kódem 250.

Autor se upřímně domnívá, že tato technologie je zabitá již při svém uveřejnění. Navíc je zatížená patentem a jen zvyšuje složitost celého procesu.

přes to poslední verze softwarových nástrojů pro práci se SPF, uveřejněné na adrese *http://spf.pobox.com/*, umějí kontrolovat kromě SPF i informace uvedené ve strukturách MS Caller-ID.

Schéma RMX++

Je součástí technologie SCAF (Simple Caller Authorization Framework). Schéma vytvořil Hadmut Danisch, který je také autorem původního RMX. Technologie RMX++ umožňuje provádět dynamickou autorizaci způsobem, který se využívá u webových serverů a protokolu HTTP. Vlastník domény zveřejní umístění autorizačního serveru v systému DNS a hostitelský systém příjemce pak kontaktuje tento autorizační server a získá od něj *autorizační záznam*, kterým ověří autenticitu odesílatele.

Toto schéma umožňuje vlastníkovi domény dokonalou kontrolu nad kritérii použitými pro auten-tizaci adresy odesílatele bez nutnosti zveřejňovat informace o struktuře sítě (jako v případě infor-mací SPF ve statických záznamech typu TXT). Příkladem přímo od Hadmuta je třeba autorizační server, který nepovolí odeslání více jak 5 zpráv z dané adresy mimo pracovní dobu a v případě překročení tohoto limitu vygeneruje varování.

Technologie SCAF není omezena pouze na zprávy elektronické pošty, ale umožní provádět auten tizaci i pro jiné služby, například VoIP. Jednou nevýhodou schématu RMX++, na kterou upozorňuje Rick Stewart *<rick.stewart@theinter netco.net>*, je vliv na výpočetní a síťové zdroje. Odezvy ze serveru HTTP nebývají tak často uklá-dány do vyrovnávacích pamětí jako informace získané přímo za systému DNS a sestavit a zaslat požadavek HTTP je daleko náročnější než provést dotaz do systému DNS.

Rick dále upozorňuje na to, že díky dynamické povaze schématu RMX++ se případné chyby vzniklé při autentizaci špatně vyhledávají. Pokud existuje limit 5 zpráv za den a jedinou zprávu je nutné díky chybám kontrolovat pětkrát, spotřebuje se celý limit na doručení jediné zprávy.

Více informací o schématu RMX, RMX++ a SCAF naleznete na adrese *http://www.danisch.de/ work/security/antispam.html.*

Kontrola obsahu zprávy

V této kapitole naleznete informace o kontrole vlastního obsahu zprávy. Takové kontroly prová-dějí klasické antivirové a antispamové programy, protože pracují se zprávou až po jejím přijetí. V našem případě budeme tyto kontroly provádět ještě před zasláním ukončovacího kódu 250, takže bude možné odmítnout doručení zprávy místo toho, abychom se stali později zdrojem zavle-čeného spamu.

V případě, že jsou poštovní servery, které zpracovávají příchozí poštu, hodně zatížené (velká orga-nizace, málo serverů), je provádění kontrol obsahu zpráv přímo na serveru nevýhodné. Poměrně značnou část výkonu serveru si také zaberou antivirové a antispamové programy. Výhodné řeše-ní je v takovém případě zřejmé, stačí pro tyto programy vyhradit zvláštní počítač. Většinu serve-rových verzí antispamového a antivirového softwaru je možné spouštět ze sítě, v tomto případě z poštovního serveru. Více informací naleznete v dalších kapitolách, kde bude podrobně popsá-na implementace systému pro různé agenty přenosu zpráv.

Kontroly záhlaví zprávy

Chybějící záhlaví

Dokument RFC 2822 udává, že zpráva by měla obsahovat minimálně tato záhlaví:

From: ...
To: ...
Subject: ...

Message-ID: ...
Date: ...

Pokud kterýkoliv řádek chybí, znamená to, že zpráva pravděpodobně nebyla vygenerovaná agentem přenosu zpráv a jedná se o nevyžádanou poštu.

> Poznámka
>
> Někteří specializovaní agenti přenosu zpráv, například některé servery mailinglistů, nege nerují záhlaví Message-ID: pro vrácené zprávy (oznámení o stavu doručení) automaticky. Takové zprávy jsou jednoznačně identifikované chybějící adresou v části MAIL FROM:.

Syntaktická kontrola adres v záhlaví

Adresy, které jsou součástí záhlaví zpráv (tj. To:, Cc:, From:), by měly být syntakticky správné.

Jednoduchá kontrola adres v záhlaví

Každá adresa v záhlaví zprávy by měla splňovat následující kritéria:

Pokud je adresa lokální, obsahuje část adresy před znakem @ existující poštovní schránku?
Pokud je adresa vzdálená, existuje doména z části adresy za znakem @?

Ověření adres ze záhlaví zpětným voláním

Tato kontrola funguje stejně jako ověření adresy odesílatele nebo ověření adresy příjemce. Každá vzdálená adresa ze záhlaví je ověřena s využitím primárního poštovního serveru (zjištěného ze záznamu typu MX) odpovídající vzdálené domény. Tím se zjistí, jestli vzdálený poštovní server může přijímat oznámení o stavu doručení.

Úložiště signatur zpráv nevyžádané pošty

Společným znakem zpráv nevyžádané pošty je, že jsou zasílány na velké množství adres. Pokud 50 jiných příjemců zprávu označilo jako nevyžádanou poštu, proč tento fakt nevyužít při zjišťová-ní, jestli danou zprávu přijmou nebo odmítnout? Proč nezajít ještě dále a nevytvořit zvláštní adre-su, tzv. spamovou past, která se stane zdrojem pro úložiště signatur zpráv nevyžádané pošty?

Jsem rád, že tyto dotazy zazněly. Jednoduchou odpovědí je, že taková úložiště již existují. Patří mezi ně například:

Razor (viz *http://razor.sourceforge.net/*).
Pyzor (viz *http://pyzor.sourceforge.net/*).
DCC (Distributed Checksum Clearinghouse, viz *http://rhyolite.com/anti-spam/dcc/*).

Tyto nástroje se postupně vyvinuly od nejjednodušších verzí, které na základě kontrolního součtu zprávy byly schopny zjistit, že přijatá zpráva je identická kopie jiné zprávy již klasifikované jako nevyžádaná pošta, až k důmyslným pozdějším verzím, které vyhodnocují společné hlavní znaky záhlaví i těla zprávy a jsou schopny odhalit a správně detekovat i drobné úpravy zpráv nevyžádané pošty.

Kontroly na binární smetí

Zprávy, které obsahují netisknutelné znaky, se vyskytují pouze zřídka. Výskyt takové zprávy s nej-větší pravděpodobností znamená virus nebo v některých jiných případech spam, který obsahuje text v nějakém exotickém jazyce bez odpovídajícího kódování MIME.

Zvláštním případem jsou zprávy, které obsahují znaky NUL (ordinální hodnota tohoto znaku je 0). I v případě, kdy se rozhodnete kontrolu netisknutelných znaků neprovádět, je vhodné ponechat alespoň kontrolu na znak NUL. Je to proto, že někteří agenti přenosu zpráv, jako je například Cyrus Mail Suite, takové zprávy automaticky odmítnou.

> Poznámka
>
> Protokol IMAP nepovoluje přenos znaků NUL na poštovního klienta uživatelů, takže se vývojáři z firmy Cyrus rozhodli, že nejjednodušší způsob, jak se vyrovnat se zprávami, které takové znaky obsahují, je odmítnout doručení.

Pokud tedy takový software používáte, je nutné tyto výjimky nějak ošetřit. Na druhou stranu, specifikace RFC 822 (dnes již zastaralá) používání znaků NUL ve zprávě explicitně nezakazuje. Proto je místo odmítnutí doručení zprávy vhodné před předáním zprávy softwaru Cyrus zajistit odstranění znaků NUL ze zprávy.

Kontroly kódování MIME

Další vhodnou kontrolou je kontrola struktury MIME v příchozích zprávách. Chyby nebo nekonzistence vzniklé při dekódování MIME nejsou příliš časté, ale pokud vzniknou, je více než pravděpodobné, že zpráva je spam. Takové chyby mohou navíc způsobit potenciální problémy při dalších kontrolách příloh zprávy nebo kontrolách antivirovým nebo antispamovým programem.

Krátce řečeno, pokud je kódování MIME chybné, zprávu je vhodné odmítnout.

Kontroly příloh zprávy

Kdy se naposledy stalo, že jste potřebovali od někoho poslat spořič obrazovky (soubor s přípo

nou .scr) nebo soubor s informacemi o programu (soubor s příponou .pif)?Je velmi vhodné zvážit blokování všech zpráv, které obsahují v příloze spustitelné soubory, tzn.soubory, jejichž název obsahuje třípísmennou příponu podobnou těm uvedeným výše. Tato kon-trola zabere znatelně méně výpočetního výkonu než antivirový skener a může také zachytit ještěneznámé viry, pro které antivirový program nemá příslušné informace ve své virové databázi.

Podrobný seznam přípon souborů naleznete na adrese *http://support.microsoft.com/ defa-ult.aspx?scid=kb;EN-US;290497.*

Antivirové programy

V současnosti jsou dostupné různé serverové antivirové programy. Mezi nejznámější patří například:

Sophie (viz *http://www.vanja.com/tools/sophie/*)
KAVDaemon (*viz http://www.kapersky.com*)
ClamAV (viz *http://www.clamav.net/*)
Dr.WEB (viz *http://www.sald.com/*)

Pokud nechcete blokovat všechny potenciálně nebezpečné soubory na základě jejich názvu (vez-měme třeba soubory .zip), jsou takové programy velmi užitečné. Programy jsou kromě jiného schopny zachytit i viry, které nejsou přenášeny v podobě příloh souborů (tak byl například pře-nášen virus Bagle.R v březnu 2004).

Systém, na kterém se provádí antivirová kontrola, nemusí být nutně poštovní server. Většinu těch to antivirových programů je možné spustit i na vzdáleném systému přes síť. Antivirový software obvykle viry vyhledává na základě signatur známých virů, tzv. *virových defi-nicí*. Tyto definice je nutné pravidelně aktualizovat současně s tím, jak vznikají nové viry. Pro co možná nejpřesnější kontrolu by antivirový software měl být vždy aktualizovaný na svou poslední verzi.

Antispamové skenery

Antispamový software se používá ke klasifikaci zpráv s využitím rozsáhlé množiny heuristických kritérií, které zahrnují obsah zprávy, dodržování standardů nebo různé síťové kontroly, například kontroly s využitím seznamů zakázaných položek systému DNS nebo úložišť signatur zpráv nevy-žádané pošty. Na konci tohoto procesu antispamový software každou zprávu ohodnotí složeným skórem, které udává pravděpodobnost, že zpráva je nevyžádaná pošta. Pokud toto skóre přesáh-ne jistou hranici, je zpráva klasifikována jako nevyžádaná pošta.

Dva nejpoužívanější serverové spamové filtry jsou:

SpamAssassin (viz *http://spamassassin.apache.org/*)
BrightMail (viz *http://www.brightmail.com/*)

Tyto nástroje procházejí neustálým vývojem, protože spameři hledají způsoby, jak různé kontroly obejít. Příkladem může být třeba „kreativní“ abeceda, která využívá podobnosti různých znaků (jeden příklad za všechny: V1/\GRA = viagra). Podobně jako u antivirového softwaru i u antispamového softwaru platí, že je nutná pravidelná aktualizace na nejnovější verzi.

Autor využívá software SpamAssassin, nicméně tento software není nasazen v první obranné linii, čímž se šetří výpočetní výkon. Z cca 500 zpráv nevyžádané pošty za den se zhruba 50 zpráv dosta-ne až do místa, kde je kontroluje software SpamAssassin (hlavně díky tomu, že autor si zprávy přeposílá ještě z jiného účtu, takže výše popsané kontroly nejsou účinné). Z tohoto množství zpráv pak v poštovní schránce autora končí zhruba jedna zpráva za 2–3 dny.

Blokování zavlečeného spamu

Zavlečený spam (collateral spam) je s využitím technik popsaných v tomto dokumentu daleko obtížněji filtrovatelný, protože pochází z legitimních poštovních serverů, které využívají standard-ní software pro přenos zpráv (např. Postfix, Sendmail nebo Exim). Velkou výzvou pak bývá roz-lišení těchto zpráv od legitimních oznámení o stavu doručení, které jsou generovány jako odezva na zprávu zaslanou jedním z vlastních uživatelů.

Filtr na falešná varování před počítačovými viry

Zavlečený spam tohoto typu obvykle pochází z varování před počítačovými viry, která jsou zasílaná antivirovými programy.

> Poznámka
>
> Důvod, proč autoři antivirového softwaru důvěřují pravosti adresy odesílatele ve zprávě, která obsahuje virus, je zřejmě námětem na podrobnější psychoanalytickou studii.

Text předmětu takových falešných varování (záhlaví *Subject:*) a ostatní charakteristiky těchto zpráv jsou vytvářeny přímo antivirovým softwarem. Proto je velmi snadné sestavit seznam společných znaků takových zpráv a odfiltrovat je.

Tuto práci za vás naštěstí udělal už někdo jiný. Seznam falešných varování před počítačovými viry udržuje Tim Jackson <*tim@timj.co.uk*>. Seznam je možné použít v softwaru SpamAssassin a je k dispozici na adrese *http://www.timj.co.uk/linux/bogus-virus-warnings.cf.*

Uveřejnění informací SPF pro doménu

Účelem záznamů SPF (Sender Policy Framework) je chránit před nevyžádanou poštou, která fal
šuje adresu odesílatele, a tudíž před zneužitím legitimních adres elektronické pošty. Pokud uveřejníte záznamy SPF v zóně DNS vaší domény, hostitelské systémy příjemce zprávy, které provádějí kontroly s využitím SPF, by neměly akceptovat zprávy s podvrženými adresami odesílatele. Stejně tak by tyto systémy neměly pro takové nedoručitelné zprávy zasílat oznámení o stavu doručení zpět do vaší domény.

Signatura odesílatele zprávy

Poněkud odlišný způsob, se kterým autor v současnosti experimentuje, využívá přidání signatury odesílatele do adresy uvedené v části MAIL FROM: a následnou kontrolu této signatury v adrese RCPT TO: před přijetím příchozího oznámení o stavu doručení. Vygenerovaná adresa odesílatele může mít například následující formát:

odesílatel=signatura@doména

Normální odpovědi na zprávy jsou zpracovány beze změny. Odpovědi se zasílají na adresu ze záhlaví *From:* nebo *Reply-To:*, která jsou přenášena beze změny.Zní to snadno, že? Naneštěstí je vygenerování signatury, která je pro tyto účely vhodná, poněkudsložitější, než to může na první pohled vypadat. Existují některé protichůdné požadavky, se kterými je nutné počítat.

- Vygenerovaná signatura adresy odesílatele by měla být v rukou spamera bezcenná. To zna-mená, že součástí signatury by mělo být časové razítko, u kterého je možné kontrolovat platnost:

  odesílatel=časové_razítko=kontrolní součet@doména

- Při zasílání zprávy do domény, která využívá greylisting, je nutné, aby adresa odesílatele zůstávala pro daného příjemce konstantní. V opačném případě bude zpráva neustále odmí-tána. Pokud budeme počítat i s tímhle, je možné vygenerovat signaturu pro odesílatele v závislosti na adrese příjemce:

  odesílatel=příjemce=doména.příjemce=kontrolní_součet@doména

  Taková adresa nemá omezenou časovou platnost, a pokud z ní začne chodit nevyžádaná pošta, bude určitě známý zdroj – je obsažen v adrese příjemce. Kromě toho je možné sig-natury specifických adres příjemců snadno blokovat bez vlivu na doručování jiných zpráv stejnému příjemci.

- Při využívání mailinglistů je možné narazit na dva problémy. Odpovědi na zprávy s poža-davkem na zařazení nebo vyřazení z mailinglistu jsou obvykle zasílány bez adresy odesí-latele.

  - První problém se týká serverů, které zasílají odezvu zpět na adresu odesílatele ze zprá-vy s požadavkem na zařazení nebo vyřazení (jako v případě <*discuss@en.tldp.org*>). Problém je, že příkazy pro server, který zajišťuje mailinglist (např. subscribe nebo unsubscribe) jsou obvykle zasílány na jiné adresy (např. <discuss-sub scribe@en.tldp.org> a <discuss-unsubscribe@en.tldp.org>), než je adresa pou-žitá pro vlastní mailinglist. V tomto případě se tedy bude adresa uživatele, který žádá o zařazení, lišit od adresy odesílatele ve zprávách zaslaných serveru mailinglistu a bude se lišit i od adres, které budou vygenerovány u požadavků na vyřazení z mailinglistu. Výsledkem může být situace, kdy není možné do mailinglistu zasílat příspěvky nebo požádat o vyřazení. Vhodným kompromisem je v tomto případě zařadit do signatury odesílatele pouze doménu příjemce. Adresa odesílatele pak může vypadat například takto:

    název_přispěvatele=en.tldp.org=*kontrolní_součet@doména.přispěvatele*

  - Druhý problém se týká těch uživatelů, kteří zasílají odpovědi přímo na adresu pro odpovědi zjištěnou ze záhlaví zprávy s požadavkem (jako např. <spam-l-re quest@peach.ease.lsoft.com>). Protože tato adresa neobsahuje signaturu, odpověď od serveru s mailinglistem může být na vašem serveru blokovaná. Tento problém se dá řešit pouze tak, že se servery s mailinglistem umístí do seznamu povolených položek tak, aby bylo umožněno vracení zpráv na adresy příjemců i bez signatury.

V souvislosti se zmíněnými problémy začíná tato technologie ztrácet dech. Dalším problémem je, že v případě, kdy původní zpráva nebyla zaslána přes váš poštovní server, jsou legitimní ozná-mení o stavu doručení odmítána. Výše uvedená opatření je proto vhodné provádět pouze pro ty uživatele, kteří nezasílají zprávy s využitím poštovních serverů mimo vaši síť.

V situaci, kdy není možné využít žádnou z výše uvedených technik, umožní signatury nejen eli-minovat zavlečený spam, ale také poučit vlastníky poštovních serverů, kteří (předpokládejme nezaviněně) takový spam generují. Pozitivním vedlejším efektem je, že poštovní servery, které využívají ověření adresy odesílatele zpětným voláním, obdrží pozitivní odpověď na ověření pouze v případě, kdy byla původní zpráva zaslána vaším poštovním serverem. Klíčové však je, že pou-žitím těchto metod se spamerům

snižuje možnost podvržení a zfalšování adres odesílatele.

Uživatelům je určitě vhodné umožnit, aby se sami rozhodli, jestli chtějí jako součást odesílané pošty signatury uvádět, a pokud se rozhodnou pro používání signatur, pak určit, které hostitelské systémy mají mít umožněno vracet zprávy bez signatury adresy. Pokud mají uživatelé vytvořený systémový účet přímo na poštovním serveru, je to například možné zajistit tak, že si uživatel vytvoří vlastní „konfigurační“ soubor přímo ve své domovské složce.

Příjem vrácených zpráv pouze pro existující uživatele

I v případě, kdy se provádí kontrola signatur adres odesílatele, existuje slabé místo, díky kterému může být umožněn příjem falešných odpovědí na zprávy. Jedná se o kontrolu zpráv, které jsou zaslané na nějaké systémové aliasy (např. *postmaster* nebo *mailer-daemon*). Platí však, že pokud takoví uživatelé nezasílají žádné zprávy, neměli by dostávat žádné odpovědi.

Zprávy je tedy možné odmítnout v případě, kdy jsou zaslané na některý ze systémových aliasů nebo pokud pro danou adresu příjemce neexistuje příslušná poštovní schránka.

Další témata k zamyšlení

V souvislosti s filtrováním zpráv v rozsáhlých systémech, při kterém se využívá časových interva-lů, je nutné vzít v úvahu následující fakta.

Více poštovních serverů příchozí pošty

Většina domén obsahuje více poštovních serverů příchozí pošty (v systému DNS využívají více záznamů typu MX). Pokud je na primárním poštovním serveru použito filtrování, které využívá časové intervaly, je nutné toto filtrování použít i u ostatních záložních poštovních serverů. Pokud by tomu tak nebylo, hostitelský systém, který odesílá zprávy, by mohl filtrování jednoduše obejít tak, že zprávy zašle na záložní poštovní server(y).

Pokud nejsou záložní servery ve vaší správě (například se využívá serverů poskytovatele připoje-ní), je vhodné zvážit, jestli je vůbec nutné uvádět v informacích o doméně v systému DNS více záznamů typu MX. Pokud celý poštovní systém funguje tak, že záložní poštovní servery pouze automaticky přeposílají všechny došlé zprávy na primární poštovní server, je možné ostatní zá-znamy typu MX v informacích o doméně odstranit. Pokud dojde k výpadku primárního poštovní-ho serveru, nic se nestane, protože hostitelské systémy odesílatelů zpráv se pokoušejí o zaslání
nedoručených zpráv dalších několik dní. Jediná situace, za které je vhodné mít několik poštovních serverů, je stav, kdy je nutné provádět vyvažování výkonu mezi více servery, tzn. že doména přijímá více zpráv, než je schopen jeden poštovní server obsloužit. V takovém případě je vhodné výpočetně náročné úlohy (antivirová a antispamová kontrola) přesunout na jiné servery, což může v důsledku znamenat, že pro příjem pošty bude stačit pouze jediný server.
Pokud se přeci jen rozhodnete využívat více poštovních serverů, je nutné, aby všechny záložní servery byly, co se filtrování zpráv týče, nakonfigurovány stejně jako primární server. Více informací o používání více poštovních serverů naleznete v kapitole Greylisting.

Blokování přístupu na jiné servery SMTP

Žádný server SMTP, který není uvedený v záznamech typu MX pro doménu v systému DNS, by neměl akceptovat příchozí připojení z Internetu. Veškerý příchozí poštovní provoz by měl jít pouze přes vyhrazený poštovní server.

Toto pravidlo neplatí pouze pro server SMTP. Pokud v síti využíváte servery, které slouží pro inter-ní potřebu, omezte přístup k takovým serverům z Internetu pomocí firewallu.

Přeposílané poštovní zprávy

Konfiguraci filtrování je nutné nastavit tak, aby nedocházelo k odmítání takových zpráv, které jsou přeposílány z takových důvěryhodných zdrojů, jako jsou například:

záložní poštovní servery. Pokud využíváte záložní server, dá se samozřejmě předpokládat, že většina nevyžádané pošty je již odfiltrovaná.
servery s mailinglisty, které odebírají vaši uživatelé. Takové zprávy je samozřejmě možné filtrovat (a určitě je to lepší přístup, než by zprávy měly skončit v černé díře). Pokud však dojde k odmítání takových zpráv, může to vést až k automatickému vyřazení daného uži-vatele z mailinglistu.

- ostatní účty uživatelů vaší domény. Odmítání takových zpráv může generovat zavlečený spam anebo způsobit problémy hostitelskému systému, který zprávy přeposílá.

U dvou posledních uvedených zdrojů vzniká logistický problém. Tyto zdroje jsou specifické pro každého příjemce ve vaší doméně. Jak zařídit, aby si každý uživatel mohl vytvořit svůj seznam povolených položek, a jak zajistit, aby se obsah jednotlivých seznamů promítl v celkové konfigu-raci filtrování SMTP například pro celou doménu? Pokud je zpráva přeposílána více

příjemcům ve vaší doméně (to může být například případ mailinglistu), jak rozhodnout, čí seznam povolených položek použít?

V tomto případě neexistuje univerzální řešení a je nutné vše řešit individuálně. Můžete se napří-klad rozhodnout pro příjem všech zpráv bez ohledu na jejich spamovou klasifikaci v případě, kdy jsou zaslány z hostitelského systému, který figuruje alespoň v jednom ze všech seznamů povole-ných položek libovolného uživatele. Odezvu na jednotlivé příkazy RCPT TO: je možné vytvářet na základě kontroly odpovídajícího hostitelského systému odesílatele s pomocí seznamu povole-ných položek příslušného uživatele. Pokud je hostitelský systém v seznamu nalezen, je možné nastavit příznak, který zabrání odmítnutí zprávy při dalších kontrolách. V takových případech je také vhodné používat výsledný seznam jako sjednocení všech seznamů povolených položek jed-notlivých uživatelů.

Podrobnosti naleznete v popisu konkrétní implementace filtrování v dodatcích.

Uživatelská nastavení a data

Existují situace, kdy je nutné uchovávat nastavení a data pro každého uživatele domény zvlášť. Pokud například využíváte pro filtrování příchozí pošty software SpamAssassin, můžete pro jed-notlivé uživatele umožnit nastavení vlastních prahových hodnot pro klasifikaci spamu, povolené jazykové a znakové sady nebo testovací data pro Bayesův filtr.

Stěžejním bodem je, že filtrování protokolu SMTP s využitím časových intervalů se provádí na systémové úrovni ještě předtím, než jsou zprávy doručeny jednotlivým uživatelům, což do jisté míry neumožní používat individuální nastavení pro jednotlivé uživatele. Jediná zpráva může mít více příjemců a na rozdíl od přeposílané zprávy není v tomto případě vhodné použít sjednocené nastavení ode všech uživatelů – adresátů zmíněné zprávy. Vezměme například v úvahu situaci, kdy jednotliví příjemci zprávy jsou z různých zemí a hovoří různými jazyky.

Výjimkou je samozřejmě případ, kdy se počet příjemců zprávy v příchozích zprávách omezí na hodnotu 1. Zprávu je za této podmínky možné analyzovat a kontrolovat na základě individuálních nastavení příslušného uživatele.

To se zajistí akceptováním prvního příkazu RCPT TO: a následným zasláním odezvy s kódem 451. Pokud je odesílatelem zprávy legitimní poštovní server, ví, jak tuto odezvu vyhodnotit, a pokusí se o další doručení později.

Samozřejmě existuje i jiný způsob, který využije zpoždění, takže u všech zpráv s vícenásobným příjemcem zpozdí doručení například o 30 minut na jednoho příjemce. V prostředí velké organi-zace však tento způsob nemusí být nejvhodnější, protože v takovém prostředí probíhají diskuse s využitím poštovních zpráv takřka v reálném čase a zpoždění zpráv by mohlo mít negativní dopa-dy např. na plánování nebo při řešení nějakých akutních situací.

Dalším problémem, který se opět týká v hlavní míře prostředí velké organizace, je, že příchozí zprávy jsou pro doručení obvykle dále předávány na interní servery a jednotliví uživatelé na poš-tovním serveru ani nemají vytvořené své uživatelské účty. Individuální nastavení pro jednotlivé uživatele se dá použít i v takovém prostředí, ale vyžaduje to dodatečné úsilí např. s využitím vyhle-dávání v databázi nebo v adresáři LDAP a je na zvážení každého, jestli takové řešení implemen-tovat.

Otázky a odpovědi

V této kapitole je uvedeno shrnutí některých otázek, které při čtení dokumentu čtenáře určitě napadnou. Pokud máte dotaz, který zde není uveden, a rádi byste v této kapitole viděli odpověď, prosím, napište na adresu <tor@slett.net>.

Když se spameři přizpůsobí

Otázka: Co se stane, když se spameři přizpůsobí a obejdou jednotlivé způsoby filtrování, popsa-né v tomto dokumentu?

Odpověď: Hm, to záleží na konkrétní situaci. Některé popsané způsoby filtrování (například kontroly protokolu SMTP nebo greylisting) jsou zaměřeny na specifické chování ratwaru. Dá se předpokládat, že chování ratwaru se určitě změní v případě, kdy bude uvedené kontroly provádět dostatečné množství poštovních serverů. Hatmut Danish uvádí: *„Ratware obsahuje neúplnou implementaci protokolu SMTP proto, že vývojáře rat-waru nic nenutí k tomu, aby implementaci zdokonalili. Pokud daná implementace funguje, proč ztrácet čas jejím zdokonalováním? Kvalita ratwaru se však postupem doby zvyšuje a stejně tak se zvyšuje i kvalita spamových zpráv. Pokud začne dostatečný počet lidí odmítat spam tak, že budou kontrolovat implementaci protokolu SMTP, autoři spamového softwaru prostě svůj software vylepší.“*

Autoři ratwaru budou muset řešit následující problémy:

- Pro překonání zpoždění transakcí protokolu SMTP budou muset čekat na každou odezvu od serveru SMTP, ke kterému se ratware připojuje. V tomto případě je pak dosaženo pod-statného snížení zpráv, které je hostitelský systém spamera schopen v daném čase odeslat. Protože spameři zápasí s časem a snaží se doručit maximum zpráv ještě předtím, než jsou odfiltrovány s využitím seznamů zakázaných položek systému DNS nebo filtrů obsahu zpráv, bude nutné zdokonalit efektivitu nástrojů, které zpoždění transakcí provádějí.
Efekt je podobný cílům, které mají tzv. schémata mikroplateb, kdy odesílatel stráví něko-lik sekund výpočtem signatury pro každého příjemce zprávy a tuto signaturu přidá do záhlaví zprávy tak, aby ji mohl příjemce ověřit. Pokud pomineme složitost takových sché-mat, je hlavním rozdílem to, že vyžadují participaci pokud možno všech uživatelů na světě. Efektivní filtrování nevyžádané pošty je pak zaručeno pouze při účasti co největšího počtu uživatelů. Zpoždění transakcí protokolu SMTP je na druhou stranu efektivní už u prvního hostitelského systému, který je implementuje.

Pro překonání kontroly příkazu HELO/EHLO musí ratware dodat korektní informace, tzn. musí se identifikovat úplným názvem domény. To umožní lepší sledování trasy zpráv zvláš-tě u takových agentů přenosu zpráv, kteří neprovádějí automatické vkládání výsledku reverzního dotazu do systému DNS do záhlaví *Received:* přenášené zprávy.
Pro překonání všech kontrol protokolu SMTP musí ratware doplňovat vlastní platnou adresu odesílatele (nebo alespoň platnou adresu odesílatele v rámci své domény).
Pro překonání greylistingu se budou muset pokoušet o opakování přenosu dočasně odmít-nutých zpráv v limitu 1 hodina (ale ne za déle než 4 hodiny). Aby se minimalizovaly vyu-žité prostředky, bude si ratware zřejmě uchovávat místo kopie každé odmítnuté zprávy pouze seznam dočasně odmítnutých příjemců a za daný časový interval se pokusí o nové hromadné doručení.

> Greylisting zůstane i v budoucnu účinný hlavně ve spojení se seznamy zakázaných polo-žek systému DNS, které jsou naplňované ze spamových pastí. Je to dáno tím, že hodino-vé zpoždění po odmítnutí zprávy umožní aktualizaci seznamu o hostitelský systém, který spam zasílá.

Všechny softwarové nástroje, antivirové i antispamové, se neustále vyvíjejí. Jejich efektivní použí
vání je zaručeno pouze v případě, kdy používáte nejnovější, aktualizované verze.Stejně tak se bude vyvíjet i tento dokument.
Spolu s tím, jak se změní povaha nevyžádané pošty,se bude měnit i přístup lidí při hledání způsobů, jak nevyžádanou poštu blokovat.

Implementace s využitím softwaru Exim

V následujících kapitolách naleznete implementaci technik blokování spamu, které jsou popsány v tomto dokumentu, s využitím agenta přenosu zpráv Exim.

Co budete potřebovat

Pro všechny uvedené příklady budete potřebovat agenta přenosu zpráv *Exim*, pokud možno s doplňkem *Exiscan-ACL* od Toma Kistnera. Předpřipravené balíčky, které obsahují *Exim* i *Exis-can-ACL*, existují pro většinu současných distribucí Linuxu a také pro FreeBSD. Podrobnosti nalez-nete na webových stránkách *Exiscan-ACL* (viz *http://duncanthrax.net/exiscan-acl/*).

> Poznámka
>
> Software Exim je oblíbený zejména mezi uživateli distribuce Debian GNU/Linux (viz *http://www.debian.org/*), protože je to výchozí agent přenosu zpráv této distribuce. Pokud používáte Debian (verzi Sarge nebo novější), můžete získat Exim a Exiscan-ACL instalací balíčku exim4-daemon-heavy:
>
> ```
> # apt-get install exim4-daemon-heavy
> ```

Příklad implementace uvedený v této kapitole ještě navíc obsahuje:

SpamAssassin (viz *http://spamassassin.apache.org/*) – oblíbený nástroj na filtrování spamu, který analyzuje obsah zpráv s využitím rozsáhlé množiny heuristických metod.
greylistd (viz *http://packages.debian.org/unstable/mail/greylistd*) – jednoduché řešení pro greylisting, které bylo vytvořeno uživateli s ohledem na software Exim.

V následujících příkladech se využívá ještě další software.

Konfigurační soubor Exim

Konfigurační soubor Eximu obsahuje na začátku globální definice (této části konfiguračního sou-boru budeme říkat *hlavní sekce*), které jsou následovány dalšími sekcemi.

> Poznámka
>
> Při použití balíčku exim4-config mají uživatelé distribuce Debian možnost rozdělit konfi-guraci Eximu do více malých částí, které jsou uloženy v jednotlivých podsložkách (např. /etc/exim4/conf.d), nebo nechat konfiguraci uloženou v jediném souboru.
>
> Při výběru první možnosti (je doporučována) je možné jednotlivé části upravené konfigu-race oddělit od výchozí konfigurace, která je součástí balíčku exim4-config, vytvořením nových souborů v příslušných podsložkách. V tomto případě není nutné upravovat již exi-stující soubory. Například je možné vytvořit soubor s názvem /etc/exim4/conf.d/acl/80_local-config_rcpt_to, kterým se definuje vlastní seznam řízení přístupu pro příkaz RCPT TO: (viz níže).

Každá sekce začíná textem:

```
begin název_sekce
```

Nejvíce konfiguračních úprav se bude provádět v části acl (tzn. za textem begin acl), ale budeme také upravovat položky v sekcích transports a routers a také hlavní sekci na začátku souboru.

Seznamy řízení přístupu ACL (Access Control List)

Exim od verze 4.XX obsahuje pravděpodobně nejsofistikovanější a nejflexibilnější mechanismus pro filtrování transakcí protokolu SMTP s využitím časových intervalů, při kterém se využívá seznamů řízení přístupu.

Seznamy řízení přístupu je možné použít pro vyhodnocení přijetí nebo odmítnutí příchozí trans-akce protokolu SMTP přímo v jednotlivých fázích, např. již od počátečního připojení vzdáleného hostitelského systému přes příkazy HELO/EHLO, MAIL FROM: nebo RCPT TO:. V konfiguraci je například možné vytvořit seznam acl_rcpt_to, s pomocí kterého se bude provádět kontrola pří-kazu RCPT TO: od vzdáleného hostitelského systému.

Každý seznam řízení přístupu sestává z posloupnosti *příkazů* neboli *pravidel*. Každý příkaz začí-ná popisem akce, to může být např. accept, warn, require, defer nebo deny, následovaným seznamem podmínek nebo dalších nastavení přináležících k příkazu. Jednotlivé příkazy jsou vyhodnocovány jeden po druhém, dokud nedojde k provedení konkrétní akce (výjimkou je pří-kaz warn). Na konci každého seznamu řízení přístupu je implicitně vložena akce deny.

Jednoduchý příkaz v seznamu acl_rcpt_to může vypadat například takto: deny

```
message = přeposílání není povoleno
!hosts = +relay_from_hosts
!domains = +local_domains : +relay_to_domains
delay = 1m
```

Pomocí této sekvence bude příkaz RCPT TO: (a tím i doručení zprávy) odmítnut v případě, kdy zpráva není doručována jedním z hostitelských systémů ze seznamu „+relay_from_hosts“, a pokud doména příjemce zprávy není ze seznamu „+local_domains“ nebo „+relay_to_domains.“ Před zasláním odezvy protokolu SMTP s kódem 550 bude server čekat 1 minutu.

Aby bylo možné provést vyhodnocení daného seznamu řízení přístupu v dané fázi transakce pro-tokolu SMTP, je nutné s daným seznamem svázat jednu ze *zásad řízení (policy control)*. Aby bylo například možné provést vyhodnocení seznamu acl_rcpt_to při zpracování příkazu RCPT TO:, je nutné v hlavní sekci konfiguračního souboru softwaru Exim (před kterýmkoliv klíčovým slo-vem begin) uvést:

```
acl_smtp_rcpt = acl_rcpt_to
```

Úplný seznam *zásad řízení* naleznete v kapitole 14 – specifikace softwaru Exim.

Rozšíření

Kromě základních seznamů řízení přístupu jsou k dispozici různá rozšíření, která zahrnují pro-měnné vyhodnocované při běhu programu, funkce pro vyhledávání, operace s řetězci nebo regu-lárními výrazy, seznamy hostitelských systémů nebo domén atd. Podrobný popis pro konkrétní verzi x.x0 (tzn. 4.20, 4.30) je možné najít v souboru spec.txt. Seznamy řízení přístupu nalezne-te v kapitole 38.

Software Exim poskytuje 20 proměnných, které je možné použít pro všeobecné účely a do kte-rých je možné přiřazovat hodnoty v příkazech ze seznamů řízení přístupu:

$acl_c0 -$acl_c9 uchovají hodnoty, které jsou perzistentní v rámci celého připojení SMTP.
$acl_m0 / $acl_m9 uchovají hodnoty při příjmu zprávy, pak jsou ale resetovány. Jsou také resetovány po provedení příkazů HELO, EHLO, MAIL a RSET.

Možnosti a nastavení

Hlavní sekce konfiguračního souboru Exim (před prvním klíčovým slovem begin) obsahuje různá makra, zásady řízení a jiná obecná nastavení. Zkusme definovat několik maker, která budou pou-žita později.

```
# Definice omezení velikosti zprávy. Tuto hodnotu využijeme v seznamu
# řízení přístupu DATA.
MESSAGE_SIZE_LIMIT = 10M


# Maximální velikost zprávy, pro kterou bude spuštěna antivirová nebo
# antispamová kontrola.
# Toto nastavení umožní snížit zatížení při zpracování velmi velkých zpráv.
MESSAGE_SIZE_SPAM_MAX = 1M

# Makro, které definuje klíč použitý při generování různých
# kontrolních součtů nebo hašovacích hodnot.
# TOTO NASTAVENÍ PROSÍM ZMĚŇTE!
SECRET = tajný-klíč
```

Zkusme objasnit některá obecná nastavení softwaru Exim:

```
# Selhání služby DNS (SERVFAIL) bude považováno za chybu při vyhledání.
# Díky tomu bude možné později odmítnout adresy odesílatele
```

```
# v neexistující doméně nebo v doméně, pro kterou neexistuje server
# služby DNS.
dns_again_means_nonexist = !+local_domains : !+relay_to_domains


# Aktivace kontroly příkazu HELO pro všechny hostitelské systémy
helo_try_verify_hosts = *


# Zrušení omezení maximálního počtu příchozích připojení, která je
# možné obsloužit najednou. Nastavení se provádí proto, aby bylo
# možné obsluhovat další nová příchozí připojení i při použití zpoždění
# transakcí protokolu SMTP
smtp_accept_max = 0


# ..až dokud není zatížení systému rovno hodnotě 10
smtp_load_reserve = 10


# Zákaz zveřejnění vlastnosti protokolu ESMTP “PIPELINING” všem
# hostitelským systémům.
# Nastavení se provádí kvůli ratwaru, který se snaží řetězit příkazy.
pipelining_advertise_hosts = :
```

Nakonec je nutné definovat vybrané zásady řízení pro 5 seznamů řízení přístupu, pomocí kterých bude možné kontrolovat všechny hlavní fáze příchozí transakce protokolu SMTP:

```
acl_smtp_connect = acl_connect acl_smtp_helo = acl_helo acl_smtp_mail
= acl_mail_from acl_smtp_rcpt = acl_rcpt_to acl_smtp_data = acl_data
```

Tvorba seznamů řízení přístupu – první verze

V sekci acl (následuje bezprostředně za příkazem begin acl) je nutné definovat příslušné sezna-my řízení přístupu. Při vytváření seznamů budou využity některé metody, popsané v předchozích kapitolách tohoto dokumentu, včetně kontrol systému DNS a kontrol protokolu SMTP. V této verzi se většina kontrol provede v seznamu acl_rcpt_to a ostatní seznamy zůstanou prázdné. Je to díky tomu, že ratware obvykle neumí vyhodnotit odmítnutí zprávy v dřívějších fázích transakce protokolu SMTP. Většina klientů ratwaru vzdá doručení zprávy až v případě, kdy dojde k odmít-nutí po zpracování příkazu RCPT TO:.

Nejprve všechny seznamy vytvoříme, použijeme je později.

Seznam acl_connect

```
# Tento seznam se použije na začátku transakce při příjmu příchozího # připojení. Kontroly se provádí pro zjištění, jestli příchozí
připojení # přijmout nebo odmítnout.
acl_connect: # V této fázi nebudeme provádět žádné kontroly. accept
```

Seznam acl_helo

```
# Tento seznam se použije pro kontrolu příkazu HELO nebo EHLO # v příchozí transakci protokolu SMTP. Kontroly se provádí pro zjištění, #
jestli transakci při zpracování příkazu HELO nebo EHLO přijmout nebo # odmítnout.

acl_helo: # V této fázi nebudeme provádět žádné kontroly. accept
```

Seznam acl_mail_from

```
# Tento seznam se použije pro kontrolu příkazu MAIL FROM:
# v příchozí transakci protokolu SMTP. Kontroly se provádí pro zjištění,
# jestli transakci po kontrole adresy odesílatele přijmout nebo odmítnout.


acl_mail_from: # Přijmout příkaz. accept
```

Seznam acl_rcpt_to

```
# Tento seznam se použije pro kontrolu příkazu RCPT TO:
# v příchozí transakci protokolu SMTP. Kontroly se provádí pro zjištění,
```

jestli transakci po kontrole adresy příjemce přijmout nebo odmítnout.

acl_rcpt_to:

# Akceptuje zprávy přijaté z lokální služby SMTP (tj. ne přes TCP/IP)# Kontroly se provádí pro prázdnou adresu odesílatele.# Akceptuje zprávy přijaté z hostitelských systémů, pro které se provádí# přeposílání (relay) zpráv.# Ověření příjemce se v tomto případě neprovádí, protože klienty# jsou v mnoha případech hloupé klientské programy, které neumí# dobře obsloužit chybové kódy protokolu SMTP.accept
hosts = : +relay_from_hosts

# Akceptuje zprávy, které byly zaslány s využitím autentizovaného # připojení z libovolného hostitelského systému. Tyto zprávy obvykle # pocházejí od hloupých klientských programů, takže je ověření # příjemce vypuštěno. accept
authenticated = *

Kontroly s využitím systému DNS ## # # Výsledky těchto kontrol jsou uloženy ve vyrovnávací paměti, takže # u vícenásobných příjemců se tato kontrola neprovádí víckrát. # # Pokud se daný hostitelský systém vyskytuje v některém ze seznamů # zakázaných položek systému DNS, je zpráva odmítnuta. # Opatrně při výběru těchto seznamů, mnoho z nich způsobuje # falešné pozitivní detekce a správci seznamů někdy nemají jasné zásady # pro odstranění položek ze seznamu.

#
deny
dnslists = dnsbl.sorbs.net : \
dnsbl.njabl.org : \
cbl.abuseat.org : \
bl.spamcop.net
message = $sender_host_address je uvedena v $dnslist_domain\
${if def:dnslist_text { ($dnslist_text)}}

# Pokud reverzn vyhledán hostitelsk駁o systému odesílatele selže # (tj. v systému DNS neexistuje reverzn záznam nebo se výsledný název # neshoduje s původn adresou IP), je zpráva odmítnuta. # deny
message = Reverzn vyhledán DNS selhalo pro $sender_host_address. !verify = reverse_host_lookup

Kontroly příkazů Hello

# Pokud vzdálený systém uvede adresu IP, je zpráva odmítnuta. # deny
message = Zpráva byla doručena ratwarem log_message = vzdálený hostitel použil v příkazu HELO/EHLO adresu IP condition = ${if isip {$sender_helo_name}{true}{false}}

# Podobn kontrola v případě, kdy vzdálený systém použije 1# některý z názvů naš domény.# deny
message = Zpráva byla doručena ratwarem log_message = vzdálený hostitel použil v příkazu HELO/EHLO adresu IP condition = ${if match_domain{$sender_helo_name}\
{$primary_hostname:+local_domains:+relay_to_domains}\ {true}{false}}

deny message = Zpráva byla doručena ratwarem log_message = vzdálený hostitel nepoužil příkaz HELO/EHLO condition = ${if def:sender_helo_name {false}{true}}

Pokud kontrola příkazu HELO selže, do těla zprávy se přid záhlav # X-HELO-Warning:.

log_message = vzdálený hostitel použil neověřitelný příkaz HELO/EHLO !verify = helo

```
#
warn
    message      = X-HELO-Warning: Vzdálený hostitel $sender_host_address \
                   ${if def:sender_host_name {($sender_host_name) }}\

                   použil neověřitelný název $sender_helo_name
```

#
Kontroly adresy odesílatele
##

# Pokud nen možn ověřit adresu odesílatele, je zpráva odmítnuta. # # Použit možnosti „callout“ je na zvž曙n každ馼o. Obecně plat, # že pokud je odchoz zpráva zasílána s využitím vyhrazen馼o systému # (tzv. smarthost), ověřen ned žádnou užitečnou informaci. # # Podrobnosti o chybném pokusu o ověřen zpětným voláním jsou # součást odezvy s k➉em 550; pokud je nutn podrobnosti vynechat, # stač změnit “sender/callout” na “sender/callout,no_details”. # deny
message = <$sender_address> zřejmě nen platn \ adresa odesílatele. !verify = sender/callout

#
Kontroly adresy příjemce
#
Odmítnut zprávy se provede, pokud jméno v adrese obsahuje znaky @ # nebo % nebo / nebo | nebo !. Tyto znaky se ve jméně vyskytuj velmi # zř宜ka, ale občas je zkouš použít uživatel, kteř se snaž # obejít omezen při přeposílán zpráv. # # Odmítnut zprávy se provede i v případě, kdy jméno začín tečkou. # Prázdn části adresy nejsou podle striktního výkladu dokumentu # RFC 2822 povoleny, ale Exim je umožňuje využít, protože se obvykle # normálně používaj. Pokud adresa začín tečkou, může to způsobit # problém v případě, kdy se jméno použív jako název souboru (například # pro mailinglist). # deny
local_parts = ^.*[@%!/|] : ^\\.

# Ukončen připojen se provede v případě, kdy je adresa odesílatele # prázdn, ale zpráva m více adres příjemců. Legitimn oznámen # o stavu doručen nen nikdy zasláno na více adres. # drop
message = Legitimn oznámen o stavu doručen nejsou zasílána\
          více příjemcům. senders = : postmaster@* condition = $recipients_count
# Odmítnut adresy příjemce se provede v případě, kdy příjemce je# z jin domény, než pro kterou se pošta zpracováv.#deny
message = Nepovolený relaying!domains = +local_domains : +relay_to_domains

# Odmítnut adresy příjemce se provede v případě, kdy adresa # nekoresponduje s existujíc poštovn schránkou. # Pokud poštovn schránka nen uložena v aktuálním systému (tj. pokud # je aktuáln systém pro naši doménu záložn), provede se ověřen zpětným # voláním. Pokud cílový server neodpov宜, adresa příjemce se i # přesto přijme. # deny
message = neznámý uživatel!verify = recipient/callout=20s,defer_ok

V ostatních případech je adresa příjemce OK.
#

accept

Seznam acl_data

Tento seznam řízení přístupu se používá pro kontrolu obsahu zprávy # přijaté protokolem SMTP. Kontroly se provádějí postupně # za sebou až do přijetí nebo odmítnutí zprávy.

acl_data:

# Přidání záhlaví Message-ID do zprávy zaslané klienty z vlastní domény # v případě, kdy záhlaví ve zprávě chybí. warn
condition = ${if !def:h_Message-ID: {1}}hosts = : +relay_from_hostsmessage = Message-ID: <E$message_id@$primary_hostname>

Akceptování zprávy přijaté z lokální služby SMTP (tj. ne přes TCP/IP)# Kontrola se provádí pro prázdnou položku odesílajícího host.
systému.# Akceptuje zprávy, přijaté z hostitelských systémů, pro které se provádí# přeposílání (relay) zpráv.accept
hosts = : +relay_from_hosts

# Akceptování zprávy v případě, kdy byla zpráva přijata přes # autentizované připojení z libovolného hostitelského systému.#accept
authenticated = *

Kontrola omezení velikosti zprávy # Odmítnutí zprávy v případě, kdy není korektní syntaxe záhlaví. # deny
#
deny
message = Velikost zprávy $message_size je větší než limit \
MESSAGE_SIZE_LIMIT
condition = ${if >{$message_size}{MESSAGE_SIZE_LIMIT}{true}{false}}

message = Zpráva nevyhovuje specifikaci standardu z RFC 2822 log_message = chyba při syntaktick kontrole záhlav zprávy !verify = header_syntax

# Odmítnut zpráv ze vzdálen駁o systému, kter neobsahuj záhlav # Message-ID nebo Date. # Někteř specializovan agenti přenosu zpráv (někter mailinglistov # servery) negeneruj záhlav Message-ID pro vrácen zprávy automaticky. # Proto je přidan kontrola na neprázdnost adresy odesílatele. # deny
message = Zpráva nevyhovuje specifikaci standardu z RFC 2822 log_message = chybějíc záhlav zprávy !hosts = +relay_from_hosts !senders = : postmaster@* condition = ${if or {{!
def:h_Message-ID:}\
{!def:h_Date:}\ {!def:h_Subject:}} {true}{false}}

# Varován v případě, kdy platn adresa odesílatele nen nalezena ani # v jednom ze záhlav “Sender:”, “Reply-To:” nebo “From:”. # warn
message = X-Sender-Verify-Failed: Nenalezena platn adresa\
odesílatele log_message = Nenalezen platný odesílatel !verify = header_sender

Akceptován zprávy. # accept

Vložení zpoždění do transakcí protokolu SMTP Jednoduchý způsob

Nejjednodušší způsob zpoždění transakcí protokolu SMTP je pomocí klíčového slova delay u kaž-dého posledního příkazu accept v každém již deklarovaném seznamu řízení přístupu takto:

accept delay = 20s

Kromě tohoto způsobu je možné přidávat dodatečná zpoždění u příkazů deny, které se používa-jí k vyhodnocení neplatných adres příjemců zprávy u seznamu acl_rcpt_to. Tím je možné zpo-malit slovníkové útoky. Například:

deny
message = neznámý uživatel
!verify = recipient/callout=20s,defer_ok,use_sender

delay = ${eval:$rcpt_fail_count*10 + 20}s

Zde je nutné poznamenat, že není důvod vkládat zpoždění v seznamu acl_data až po přijetí zprá-vy. Ratware se obvykle v tomto okamžiku odpojí bez čekání na odezvu od poštovního serveru. Jest-li se však klient v tomto okamžiku odpojí nebo neodpojí, nemá vliv na to, jestli Exim provede doru-čení zprávy nebo ne.

Proměnná zpoždění

Pokud jste jako já, určitě budete chtít při filtrování implementovat proměnná zpoždění, jejichž délka závisí na hostitelském systému, který provádí transakci protokolu SMTP. Jak již bylo uve-deno dříve v tomto dokumentu, je možné se rozhodnout například tak, že nalezení hostitelského systému v seznamu zakázaných položek systému DNS nebo nemožnost ověřit údaje v příkazu HELO/EHLO ještě neznamenají odmítnutí zprávy, ale jsou to dostatečné důvody pro zpoždění transakce protokolu SMTP.

Aby bylo možné implementovat proměnná zpoždění, je nutné přesunout některé kontroly, které byly součástí seznamu acl_rcpt_to, do dřívějších částí transakce protokolu SMTP. Takovým způ-sobem je možné aktivovat zpoždění okamžitě po zjištění nějakého problému a tím zvýšit pravdě-podobnost způsobení chyby synchronizace v případě, kdy zprávu zasílá ratware.

Přesuny je možné provést takto:

Kontroly s využitím systému DNS je možné přesunout do seznamu acl_connect.
Kontroly příkazu Hello je možné přesunout do seznamu acl_helo. Jedinou výjimkou je, že v tomto okamžiku není možné provést kontrolu na chybějící příkaz HELO/EHLO, pro-tože seznam se zpracovává jako *odezva* na příkaz HELO/EHLO. Tato kontrola se proto provede v seznamu acl_mail_from.
Přesunout kontroly adresy odesílatele do seznamu acl_mail_from.

Z výše uvedených důvodů není vhodné odmítnout zprávu před příkazem RCPT TO:. Místo toho se provede konverze příkazů deny na příkazy warn a pro uložení chybových zpráv a varování se využijí obecné proměnné seznamů řízení přístupu softwaru Exim. Zbytek akcí se provede v ob-sluze příkazu RCPT TO:. Vše se provede takto:

- Při rozhodnutí o odmítnutí doručení se chybová zpráva, která se využije v následné odez-vě s chybovým kódem 550, uloží v proměnných $acl_c0 nebo $acl_m0:

Pokud se zjistí podmínka pro odmítnutí ještě před doručením vlastní zprávy (tzn. v seznamech acl_connect nebo acl_helo), použije se proměnná $acl_c0. Tato pro-měnná je perzistentní a její obsah se uchovává po celou dobu trvání připojení.
Poté co začne transakce přenosu zprávy (tzn. po příkazu MAIL FROM:), provede se zkopírování obsahu proměnné $acl_c0 do proměnné $acl_m0 (její obsah se uchová-vá pouze v průběhu zpracování zprávy) a od této chvíle se opět začne používat pro-měnná $acl_c0. Díky tomu podmínky, zpracovávané v dané zprávě, neovlivní násle-dující zprávy, přijaté v rámci stejného připojení.

Podobným způsobem se provede uložení *logovacích zpráv* do proměnných $acl_c1 nebo $acl_m1.

- Pokud dojde ke zpracování podmínky, která nezaručuje okamžité odmítnutí zprávy, pro-vede se pouze uložení zprávy (varování) do proměnné $acl_c1 nebo $acl_m1. Po spuš-tění transakce přenosu zprávy (tj. v seznamu acl_mail_from) se obsah této proměnné přidá do záhlaví zprávy.

Pokud dojde k akceptování zprávy bez ohledu na výsledky jakýchkoliv dalších kontrol (například s využitím softwaru SpamAssassin), je možné nastavit odpovídající příznak do proměnných $acl_c0 nebo $acl_m0 a proměnné $acl_c1 a $acl_m1 vynulovat.
Na začátku každého seznamu řízení přístupu až do seznamu acl_mail_from včetně se do proměnné $acl_m2 provede uložení aktuálního času. Na konci každého seznamu řízení přístupu se obsah proměnné $acl_c1 nebo $acl_m1 použije pro aktivaci zpoždění trans-akce protokolu SMTP o délce trvání 20 sekund.

Použití proměnných shrnuje následující tabulka.

Použití proměnných v seznamech řízení přístupu

Proměnné $acl_[cm]0 nenastavena $acl_[cm]0 nastavena

$acl_[cm]1nenastavena (zatím žádné rozhodnutí) Přijetí zprávy $acl_[cm]1nastavena Varování do záhlaví Odmítnutí zprávy

Pro ilustraci tohoto přístupu budou uvedeny dva příklady kontroly příkazu Hello. První z nich způsobí odmítnutí zprávy v případě, kdy je součástí příkazu Hello adresa IP a druhá způsobí varo-vání o neověřitelném názvu v příkazu Hello. Dříve se obě kontroly prováděly v seznamu acl_rcpt_to. Teď budou přesunuty do seznamu acl_helo.

acl_helo: # Zaznamená aktuální čas pro výpočet doby trvání zpoždění. warn
set acl_m2 = $tod_epoch

Akceptování zprávy přijaté z lokální služby SMTP (tj. ne přes TCP/IP)# Kontrola se provádí pro prázdnou položku odesílajícího host. systému.#

Akceptuje zprávy přijaté z hostitelských systémů, pro které se provádí# přeposílání (relay) zpráv.#accept
hosts = : +relay_from_hosts

Pokud vzdálený hostitelský systém použije v příkazu Hello adresu IP, # do proměnné $acl_c0 se uloží zpráva o odmítnutí a do proměnné $acl_c1 # se uloží zpráva pro soubor protokolu. Hodnoty těchto proměnných se # využijí později při odmítnutí zprávy. V mezičase jejich nastavení # znamená, že je nutné transakci zpozdit.

#
warn
condition = ${if isip {$sender_helo_name}{true}{false}}
set acl_c0 = Zpráva byla doručena ratwarem

set acl_c1 = vzdálený hostitel použil v příkazu HELO/EHLO adresu IP

Pokud selže ověřen příkazu HELO, ulož se zpráva s varováním # do proměnn acl_c1. Toto varován bude později vloženo do záhlav # zprávy. V mezičase jejich nastaven znamen, že je nutn transakci # zpozdit.

#
warncondition = ${if !def:acl_c1 {true}{false}}!verify = heloset acl_c1 = X-HELO-Warning:
Vzdálený hostitel $sender_host_address \
${if def:sender_host_name {($sender_host_name) }}\ se nesprávně
prezentoval jako $sender_helo_name log_message = vzdálený hostitel
použil neověřitelný příkaz HELO/EHLO

#
... dalš kontroly, kter v tomto příkladu nejsou použity ...
#

Akceptuje připojen, ale pokud je v proměnn $acl_c1 uložena zpráva,
odesílatel bude zpožděn na celkovou dobu 20 sekund.
accept

set acl_m2 = ${if def:acl_c1 {${eval:20 + $acl_m2 - $tod_epoch}}{0}} delay = ${if >{$acl_m2}{0}{$acl_m2}{0}}s

Pak se v seznamu acl_mail_from provede přesun zpráv z proměnných $acl_c{0,1} do $acl_m{0,1}. Kromě toho se obsah proměnné $acl_c1 přidá do záhlaví zprávy.
acl_mail_from: # Zaznamená aktuální čas pro výpočet doby trvání zpoždění. warn set acl_m2 = $tod_epoch

# Akceptování zprávy přijaté z lokální služby SMTP (tj. ne přes TCP/IP)# Kontrola se provádí pro prázdnou položku odesílajícího host. systému.# Akceptuje zprávy přijaté z hostitelských systémů, pro které se provádí# přeposílání (relay) zpráv.#accept
hosts = : +relay_from_hosts

Proměnné $acl_c0 and $acl_c1 obsahují zprávy o odmítnutí nebo # varování, která je nutné použít při všech dalších pokusech o doručení# v rámci této transakce protokolu SMTP. Hodnoty je nutné uložit do # odpovídajících proměnných $acl_m{0,1} a vložit varování z proměnné# $acl_m1 do záhlaví zprávy. (V případě odmítnutí zprávy proměnná# $acl_m1 obsahuje zprávu pro soubor protokolu, nicméně to nevadí,# protože záhlaví bude odstraněno spolu s celou zprávou).#warn
set acl_m0 = $acl_c0set acl_m1 = $acl_c1message = $acl_c1
#

```
    # ... další kontroly, které v tomto příkladu nejsou použity ...
    #

# Akceptuje připojení, ale pokud je v proměnné $acl_c1 uložena zpráva,
    # odesílatel bude zpožděn na celkovou dobu 20 sekund.
    accept
set acl_m2 = ${if def:acl_c1 {${eval:20 + $acl_m2 - $tod_epoch}}{0}}
       delay = ${if >{$acl_m2}{0}{$acl_m2}{0}}s
```

Všechny příslušné úpravy jsou součástí konečné verze seznamů řízení přístupu, které naleznete v dalších kapitolách.

Podpora greylistingu

Existuje více možností, jak implementovat podporu greylistingu v softwaru Exim. V následujících kapitolách popíšeme některé z nich.

Démon greylistd

Greylistd je implementace, která využívá Python. Tato implementace bude použita ve výsledných seznamech řízení přístupu. Greylistd pracuje jako samostatný démon a není závislý na externí databázi. Data démona greylistd jsou ukládána v podobě 32bitových hašovaných dat. Instalaci je možné nalézt na adrese *http://packages.debian.org/unstable/mail/greylistd.* Uživatelé distribuce Debian mohou využít při instalaci APT:

```
# apt-get install greylistd
```

Pro kontrolu s využitím démona *greylistd* vložíme do seznamu acl_rcpt_to dva příkazy přímo před poslední příkaz accept. # Pro získání stavu pro daný triplet (hostitel, odesílatel, příjemce) se # využije démon „greylistd“. # # Greylisting se neprovádí pro zprávy bez odesílatele, protože ověření # odesílatele zpětným voláním by nefungovalo (a pravděpodobně by nebylo # možné zaslat zprávu systému, který toto ověřování provádí). # defer message = $sender_host_address ještě není autorizován pro doručení\ zprávy z adresy <$sender_address> na adresu\ <$local_part@$domain>. Prosím zkuste později. log_message = blokováno greylistingem. domains = +local_domains : +relay_to_domains !senders = : postmaster@* set acl_m9 = $sender_host_address $sender_address $local_part@$domain set acl_m9 = ${readsocket{/var/run/greylistd/socket}{$acl_m9}{5s}{}{}} condition = ${if eq {$acl_m9}{grey}{true}{false}}

Pokud se nebude využívat signatury adresy odesílatele pro blokování falešných oznámení o stavu doručení, je možné přidat podobnou sekvenci příkazů i do seznamu acl_data pro provedení grey-listingu pro zprávy s prázdným odesílatelem.

Data, která použijeme pro účely greylistingu, budou v takovém případě jiná než v předchozím pří-padě. Kromě toho, že proměnná $sender_address zůstane prázdná, nebudou definovány ani hod-noty proměnných $local_part a $domain. Místo toho je možné použít proměnnou $recipi-ents, která obsahuje seznam všech jednotlivých příjemců zprávy. Legitimní oznámení o stavu doru-čení by mělo obsahovat pouze jednu adresu.

```
    # Greylisting pro zprávy bez adresy odesílatele.
    # Greylisting se v tomto případě neprování po příkazu RCPT TO:, protože
    # by nefungovalo ověření odesílatele zpětným voláním ze vzdálených
    # systémů.
    #
    defer

        message = $sender_host_address ještě není autorizován pro zasílání\ oznámení o stavu doručení pro <$recipients>. \ Prosím zkuste
                        později.
        log_message = blokováno greylistingem.senders = : postmaster@*set acl_m9 = $sender_host_address $recipientsset acl_m9 = $
        {readsocket{/var/run/greylistd/socket}{$acl_m9}{5s}{}{}}condition = ${if eq {$acl_m9}{grey}{true}{false}}
```

Implementace s využitím databáze MySQL

Autorem následující implementace je Johannes Berg <*johannes@sipsolutions.net*>. Implementace využívá částečně:

Práci Ricka Stewarta <*rick.stewart@theinternetco.net*>, kterou naleznete na adrese *http://theinternetco.net/projects/exim/greylist.* Implementaci s využitím databáze Postgres od Tollefa Fog Heena <*tfheen@raw.no*>, kte-rou naleznete na adrese *http://raw.no/personal/blog/tech/Debian/2004-03-14-15-55_grey-listing.*

Implementace nevyžaduje žádné externí programy, je založena na následujících skriptech a data-bázi MySQL. Archiv s aktuální verzí skriptů a soubor README naleznete na adrese *http:// johannes.sipsolutions.net/wiki/Projects/exim-greylist.*

V systému musí být instalována databáze MySQL. V příkazovém řádku databáze MySQL se pro-vede vytvoření databáze *exim4* se dvěma tabulkami *exim_greylist* a *exim_greylist_log* následujícím skriptem:

```
CREATE DATABASE exim4; use exim4;

CREATE TABLE exim_greylist ( id bigint(20) NOT NULL auto_increment, relay_ip varchar(80) default NULL, sender varchar(255) default
    NULL, recipient varchar(255) default NULL, block_expires datetime NOT NULL default '0000-00-00 00:00:00', record_expires datetime
    NOT NULL default '9999-12-31 23:59:59', create_time datetime NOT NULL default '0000-00-00 00:00:00', type
    enum('AUTO','MANUAL') NOT NULL default 'MANUAL', passcount bigint(20) NOT NULL default '0', blockcount bigint(20) NOT
    NULL default '0', PRIMARY KEY (id)
);

CREATE TABLE exim_greylist_log ( id bigint(20) NOT NULL auto_increment,
    listid bigint(20) NOT NULL,
    timestamp datetime NOT NULL default '0000-00-00 00:00:00',
    kind enum('deferred', 'accepted') NOT NULL,
    PRIMARY KEY (id)

);
```

V hlavní sekci konfiguračního souboru agenta Exim je nutné deklarovat následující makra:

```
# definice databáze hide mysql_servers = localhost/exim4/uživatel/heslo

# nastavení # hodnoty těchto nastavení musí být platné hodnoty (xxx) pro použití # v příkazu MySQL DATE_ADD(..,INTERVAL xxx) #
Neplatné jsou hodnoty v jednotném čísle: "2 HOUR" místo "2 HOURS" GREYLIST_INITIAL_DELAY = 1 HOUR
GREYLIST_INITIAL_LIFETIME = 4 HOUR GREYLIST_WHITE_LIFETIME = 36 DAY GREYLIST_BOUNCE_LIFETIME = 0 HOUR

# zde je možné změnit názvy tabulek GREYLIST_TABLE=exim_greylist
GREYLIST_LOG_TABLE=exim_greylist_log

# dočasná deaktivace greylistingu se provede zakomentováním následujícího # řádku GREYLIST_ENABLED=

# odkomentováním následujícího řádku se aktivuje logování #GREYLIST_LOG_ENABLED=

# pod tímto komentářem již není nutné upravovat nic

.ifdef GREYLIST_ENABLED # databázová makra GREYLIST_TEST =
SELECT CASE \
    WHEN now() > block_expires THEN "accepted" \
    ELSE "deferred" \
  END AS result, id \
  FROM GREYLIST_TABLE \
  WHERE (now() < record_expires) \

                AND (sender = '${quote_mysql:$sender_address}' \ OR
                        (type='MANUAL' \ AND ( sender IS NULL \ OR sender = '$
                        {quote_mysql:@$sender_address_domain}' \ ) \ ) \ ) \ AND
                        (recipient = '${quote_mysql:$local_part@$domain}' \ OR (type =
                        'MANUAL' \
                  AND ( recipient IS NULL \ OR recipient = '${quote_mysql:$local_part@}' \ OR recipient = '${quote_mysql:@$domain}' \
                        ) \
                ) \

                    ) \ AND (relay_ip = '${quote_mysql:
            $sender_host_address}' \ OR (type='MANUAL' \ AND
            ( relay_ip IS NULL \ OR relay_ip = substring('$
            {quote_mysql:$sender_host_address}',
            1,length(relay_ip)) \ ) \ ) \ ) \ ORDER BY result DESC
            LIMIT 1

    GREYLIST_ADD = INSERT INTO GREYLIST_TABLE \ (relay_ip, sender, recipient,
        block_expires, \ record_expires, create_time, type) \
  VALUES ( '${quote_mysql:$sender_host_address}', \
    '${quote_mysql:$sender_address}', \
    '${quote_mysql:$local_part@$domain}', \
    DATE_ADD(now(), INTERVAL GREYLIST_INITIAL_DELAY), \
```

```
    DATE_ADD(now(), INTERVAL GREYLIST_INITIAL_LIFETIME), \
    now(), \
    'AUTO' \

)

GREYLIST_DEFER_HIT = UPDATE GREYLIST_TABLE \ SET blockcount=blockcount+1 \ WHERE id = $acl_m9

GREYLIST_OK_COUNT = UPDATE GREYLIST_TABLE \ SET passcount=passcount+1 \
                    WHERE id = $acl_m9

                GREYLIST_OK_NEWTIME = UPDATE GREYLIST_TABLE \ SET
                                      record_expires = DATE_ADD(now(),
                                      INTERVAL
                                      GREYLIST_WHITE_LIFETIME) \
                                      WHERE id = $acl_m9 AND
                                      type='AUTO'

               GREYLIST_OK_BOUNCE = UPDATE GREYLIST_TABLE \ SET
                                    record_expires = DATE_ADD(now(),
                                    INTERVAL
                                    GREYLIST_BOUNCE_LIFETIME) \
                                    WHERE id = $acl_m9 AND type='AUTO'

GREYLIST_LOG = INSERT INTO GREYLIST_LOG_TABLE \ (listid, timestamp, kind) \
                VALUES ($acl_m9, now(), '$acl_m8')
.endif
```

V sekci seznamů řízení přístupu (v části po begin acl) je nutné deklarovat nový seznam řízení přístupu s názvem greylist_acl. .ifdef GREYLIST_ENABLED # výsledkem vyhodnocení tohoto seznamu řízení přístupu je bu přijetí nebo # odmítnutí zprávy # Použití celého bloku závisí na hodnotě GREYLIST_ENABLED # akceptování zprávy znamená, že podmínka je TRUE a doručení se odkládá # odmítnutí zprávy znamená, že podmínka je FALSE a doručení se neodkládá
greylist_acl: # Pro normální doručení zprávy se provádí greylisting. # Kontrola tupletu, v proměnné acl_m8 vrací "accepted", "deferred" nebo # "unknown" a v proměnné acl_m9 identifikátor záznamu

```
    warn set acl_m8 = ${lookup mysql{GREYLIST_TEST}{$value}{result=unknown}} # proměnná acl_m8 = "result=x id=y"

            set acl_m9 = ${extract{id}{$acl_m8}{$value}{-1}} # proměnná acl_m9 obsahuje identifikátor záznamu nebo -1

            set acl_m8 = ${extract{result}{$acl_m8}{$value}{unknown}} # proměnná acl_m8 obsahuje
            unknown/deferred/accepted

    # kontrola existence určitého tripletu, odložení doručení, pokud
    # triplet neexistuje
    accept

            # pokud předchozí kontrola vrátila „unknown"
            condition = ${if eq{$acl_m8}{unknown}{1}}
            # přidání záznamu do databáze
            condition = ${lookup mysql{GREYLIST_ADD}{yes}{no}}


    # logování bez ohledu na výsledek
    # pokud triplet nebyl nalezen, není nutné provádět logování,
    # protože se to děje implicitně při vytvoření záznamu o času
    .ifdef GREYLIST_LOG_ENABLED
    warn condition = ${lookup mysql{GREYLIST_LOG}}
    .endif


    # kontrola, jestli je triplet stále blokován
    accept
            # pokud předchozí kontrola vrátila „deferred", pak se odkládá
            condition = ${if eq{$acl_m8}{deferred}{1}}
            # a tato akce se zaznamená
```

```
        condition = ${lookup mysql{GREYLIST_DEFER_HIT}{yes}{yes}}


    # použití klíčového slova warn pro zjištění počtu záznamů
    warn condition = ${lookup mysql{GREYLIST_OK_COUNT}}


    # použití klíčového slova warn pro nastavení nového času vypršení# platnosti u automaticky vytvářených záznamů, ale pouze v případě,# kdy
    zpráva nebyla oznámení o stavu doručení. V opačném případě se čas# nastaví na hodnotu now().warn !senders = : postmaster@*
            condition = ${lookup mysql{GREYLIST_OK_NEWTIME}}
        warn senders = : postmaster@*
        condition = ${lookup mysql{GREYLIST_OK_BOUNCE}}


    deny .endif
```

Tento seznam řízení přístupu je nutné zahrnout do seznamu acl_rcpt_to pro provádění greylis-tingu tripletů v případě, kdy adresa odesílatele není prázdná. To umožní provést ověření odesílate-le zpětným voláním:

```
.ifdef GREYLIST_ENABLED
    defer !senders = : postmaster@*
            acl = greylist_acl
            message = použit greylisting – pokuste se o doručení později .endif
```

Tento seznam se také zahrne do bloku acl_data, ale pouze v případě, kdy je adresa odesílate-le prázdná. Tím se zabrání, aby spamer obešel greylisting jednoduše tím, že nastaví prázdnou adresu odesílatele.

```
.ifdef GREYLIST_ENABLED
    defer senders = : postmaster@*
            acl = greylist_acl
            message = použit greylisting – pokuste se o doručení později .endif
```

Přidání kontrol pro SPF (Sender Policy Framework)

V této kapitole naleznete dva různé způsoby kontroly záznamů SPF s využitím agenta přenosu zpráv Exim. Kromě těchto explicitních mechanismů je více sofistikovaných kontrol záznamů SPF s přiřazováním ohodnoceného skóre dle výsledků kontrol SPF součástí softwaru SpamAssassin.

I když by *bylo* možné provádět tyto kontroly již v seznamu acl_mail_from, je zde problém, který toto rozhodnutí ovlivňuje: SPF není kompatibilní s klasickým přeposíláním zpráv. Pokud hostitel-ský systém, který zprávu přeposílá, nevyužívá nějakou implementaci SRS, může vše skončit odmít-nutím přeposílané zprávy, protože pochází z hostitelského systému, který není dle záznamů SPF autorizovaný pro přeposílání zpráv z domény, která je součástí adresy odesílatele v příkazu MAIL FROM:. Tomuto problému se zabrání tak, že se provede dodatečná kontrola uživatelských sezna-mů hostitelských systémů, ze kterých je možné přeposílané zprávy přijímat. Toto je možné pro-vést až po provedení příkazu RCPT TO:, když už známe uživatelské jméno příjemce. Proto tuto kontrolu přidáme před greylisting anebo před poslední příkaz accept v seznamu acl_rcpt_to.

Kontrola SPF s využitím doplňku Exiscan-ACL

Poslední verze doplňku Exiscan-ACL od Toma Kistnera obsahují nativní podporu pro SPF. Použi-tí je velmi jednoduché. Stačí přidat podmínku spf, kterou je možné vyhodnocovat s libovolným klíčovým slovem pass, fail, *softfail*, *none*, *neutral*, *err_perm* nebo *err_temp*. Následující kód je nutné vložit ještě před implementací greylistingu anebo před poslední příkaz accept:

```
    # Dotaz na informace SPF pro doménu odesílatele zprávy.
    # Podle výsledku se rozhodne, jestli je hostitelský systém odesílatele
    # zprávy autorizovaný pro doručení zprávy.
    # Pokud ne, je zpráva odmítnuta.
    #
    deny
                                        message = [SPF] $sender_host_address nemá umožněno zasílat zprávy \
                                z domény $sender_address_domain
        log_message = kontrola SPF selhala
        spf = fail
      # Přidání záhlaví SPF-Received: do zprávy
      warn
      message = $spf_received
```

Tento kód provede odmítnutí zprávy v případě, že vlastník domény z adresy odesílatele nepovo-lil doručování zpráv z

hostitelského systému, který zprávu zasílá. Tím se dává vlastníkům domén do rukou „moc“ rozhodovat o tom, kdo může odesílat zprávy z jejich domény. Doporučovaná alternativa je kombinovat kontroly SPF s ostatními kontrolami, například s ověřením adresy ode-sílatele zpětným voláním (stejně tak jako u předchozích kontrol platí, že nemá cenu tyto kontro-ly provádět v případě, kdy se provádí zasílání zpráv s využitím vyhrazeného hostitelského systé-mu, tzv. smarthost).

```
# Odmítnutí zprávy v případě, kdy není možné adresu odesílatele ověřit
# s využitím zpětného volání a pokud informace SPF pro doménu odesílatele
   # neumožňují hostitelskému systému provádět zasílání zpráv.
   #
   deny
      message = Adresa odesílatele zřejmě není platná, informace SPF \
                                                  neumožňují systému $sender_host_address \
      provádět zasílání zpráv z domény $sender_address_domain
      log_message = kontrola SPF selhala
      !verify = sender/callout,random,postmaster
      !spf = pass


    # Přidání záhlaví SPF-Received: do zprávy
    warn
    message = $spf_received
```

Kontroly s využitím balíčku Mail::SPF::Query

Balíček Mail::SPF::Query je oficiální testovací sada pro SPF, kterou naleznete na adrese *http://spf.pobox.com/downloads.html*. Uživatelé distribuce Debian si mohou nainstalovat balíček libmail-spf-query-perl.

Balíček Mail:SPF:Query obsahuje démona spfd, který naslouchá požadavkům v rámci domény systému UNIX. Součástí však není init-skript, který by zajistil automatické spuštění démona. V následujícím příkladu proto bude použit samostatný nástroj spfquery.

Stejně jako u předchozího typu kontrol SPF je nutné i tento kód vložit před kontroly s využitím greylistingu anebo před poslední příkaz accept v seznamu acl_rcpt_to: # Pro zjištění stavu SPF daného hostitelského systému se použije # “spfquery”. Pokud je návratová hodnota tohoto příkazu 1, # znamená to, že odesílatel není autorizovaný. # deny message = [SPF] $sender_host_address nemá povoleno zasílat zprávy \ z domény $sender_address_domain. log_message = kontrola SPF selhala set acl_m9 = -ipv4=$sender_host_address \ -sender=$sender_address \ -helo= $sender_helo_name set acl_m9 = ${run{/usr/bin/spfquery $acl_m9}} condition = ${if eq {$runrc}{1}{true}{false}}

Kontroly kódování MIME a typů souborů

Tyto kontroly závisejí na vlastnostech doplňku *Exiscan-ACL* Toma Kistnera. Exiscan-ACL obsahu-je podporu pro dekódování MIME a kontroly přípon názvů souborů. Tato kontrola zablokuje vět-šinu virů u systémů Windows, ale ne ty, které jsou přenášeny v souborech typu .ZIP, nebo tako-vé, které využívají jinou zranitelnost, například poštovního klienta Outlook.

Tyto kontroly je nutné zahrnout do seznamu acl_data před poslední příkaz accept. # Odmítnutí zpráv, které obsahují vážné chyby kódování MIME. # deny message = Zjištěna chyba kódování MIME ($demime_reason) demime = * condition = ${if >{$demime_errorlevel}{2}{1}{0}}

```
   # Rozbalení kontejnerů v kódování MIME a odmítnutí souborů s příponami,
   # které využívají viry.
   # Zde se provádí další volání příkazu demime, které ovšem vrátí výsledky
   # uložené ve vyrovnávací paměti.
   # Seznam přípon nemusí být úplný.
   #
   deny

      message = Zde neakceptujeme přílohy s příponou “.$found_extension”. demime =
      bat:btm:cmd:com:cpl:dll:exe:lnk:msi:pif:prf:reg:scr:vbs:url
```

Podmínka demime se při provádění těchto příkazů volá dvakrát. Zpráva ale ve skutečnosti není zpracovávána dvakrát, protože výsledky jsou již uloženy ve vyrovnávací paměti.

Přidání antivirového softwaru

Doplněk Exiscan-ACL umí přímo pracovat s velkým počtem antivirových programů nebo s jakým-koliv jiným antivirovým programem, který je možné spustit z příkazového řádku. Při použití anti-virového softwaru je nutné, aby hlavní sekce konfiguračního souboru softwaru Exim udávala, který antivirový program se bude používat, a také další konfigurační nastavení, platná pro daný typ antivirového programu. Základní syntaxe je:

av_scanner = typ:parametr1:parametr2:...

Například:

av_scanner = sophie:/var/run/sophie av_scanner = kavdaemon:/opt/AVP/AvpCtl av_scanner = clamd:127.0.0.1 1234 av_scanner = clamd:/opt/clamd/socket av_scanner = cmdline:/path/to/sweep -all -rec -archive %s:found:'(.+)' ...

V seznamu acl_data se pak pro antivirové testování využije podmínka *malware* takto: deny message = Tato zpráva obsahuje virus ($malware_name) demime = * malware = */defer_ok

Další informace o použití naleznete v souboru exiscan-acl-spec.txt.

Kontrola s využitím nástroje SpamAssassin

Kontrolu s využitím softwaru SpamAssassin je v prostředí softwaru Exim v průběhu transakce protokolu SMTP možné provádět dvěma způsoby:

Podmínkou spam v doplňku Exiscan-ACL. Tento způsob bude popsán dále.
S využitím nástroje *SA-Exim*, který vytvořil Marc Merlins <*marc@merlins.org*> přímo pro účely kontroly zpráv s využitím softwaru SpamAssassin. Tento nástroj využívá rozhraní Eximu local_scan() buď přímo úpravou zdrojového kódu Eximu nebo přes Marcův vlastní zásuv-ný modul dlopen(), který je součástí balíčků pro distribuci Debian exim4-daemon-light a exim4-daemon-heavy. Nástroj SA-Exim implementuje kromě jiného i greylisting a teergru-bing. Protože kontrola se v tomto případě může provést, až jsou k dispozici data obsahu zprávy, nejsou tyto kontroly tak užitečné, jako by byly v dřívějších fázích transakce proto-kolu SMTP. Nástroje SA-Exim naleznete zde: *http://marc.merlins.org/linux/exim/sa.html.*
Spuštění nástroje SpamAssassin pomocí doplňku Exiscan-ACL.

Podmínka spam umožní předat zprávu buď nástroji SpamAssassin nebo BrightMail a odpovídají-cím způsobem se zachovat v případě, kdy je zpráva klasifikována jako nevyžádaná pošta. Při vyhodnocení se ve výchozím nastavení provede připojení k démonu SpamAssassin (spamd) na místním hostitelském systému. Adresu hostitelského systému a port je možné změnit nastavením parametru spamd_address v hlavní sekci konfiguračního souboru Exim. Více informací nalezne-te v souboru exiscan-acl-spet.txt, který je součástí dokumentace doplňku.

V dalším příkladu se bude provádět odmítnutí všech zpráv klasifikovaných jako spam. Obvykle je však vhodné uchovat kopii takových zpráv ve zvláštní poštovní schránce, alespoň dočasně. Díky tomu mohou uživatelé obsah této schránky prohlížet a hledat případné legitimní zprávy, které byly klasifikovány chybně.

Exim pro tyto případy nabízí akce, které je možné aplikovat na přijatou zprávu (například free-ze). Doplněk Exiscan-ACL přidává další akci s názvem fakereject. Ta způsobí následující ode-zvu protokolu SMTP:

550-FAKEREJECT id= id_zprávy
550-Vaše zpráva byla odmítnuta, ale je uložena pro další vyhodnocení.
550 Pokud to byla legitimní zpráva, může být doručena příjemci(cům).

Tato vlastnost bude použita v následujících příkladech. Skript je nutné vložit do seznamu acl_data před poslední příkaz accept:

```
# Volání nástroje SpamAssassin pro získání hodnot $spam_score
# a $spam_report.
# V závislosti na klasifikaci obsahuje proměnná $acl_m9 hodnotu
# "ham" nebo "spam".
#
# Pokud je zpráva klasifikována jako spam, je odmítnuta.
#
warn
    set acl_m9 = ham
    spam = mail
    set acl_m9 = spam
    control = fakereject
    logwrite = :reject: Odmítnutý spam (score $spam_score): $spam_report

#                Přidání                záhlaví                X-Spam-Status:                do                zprávy.
#
```

```
                                warn
  message       = X-Spam-Status: \
                  ${if eq {$acl_m9}{spam}{Yes}{No}} (score $spam_score)\
                  ${if def:spam_report {: $spam_report}}
  logwrite      = :main: Klasifikováno jako $acl_m9 (score $spam_score)
```

V tomto příkladu je hodnota proměnné $acl_m9 nastavena na „ham“. Pak se provede volání nástroje SpamAssassin pod účtem uživatele *mail*. Pokud je zpráva klasifikována jako spam, pro-měnná $acl_m9 obsahuje hodnotu „spam“ a použije se odezva fakereject. Nakonec je do zprá-vy přidáno záhlaví *X-Spam-Status:*. Cílem je, aby klientský systém nebo agent pro doručení zprávy mohl toto záhlaví využít pro uložení zpráv s nevyžádanou poštou do zvláštní složky.

Konfigurace nástroje SpamAssassin

Výsledky vyhodnocení nástroj SpamAssassin ve výchozí konfiguraci vrací v poměrně podrobném formátu v podobě tabulky, která je vhodná pro přidání k textu zprávy nebo pro uložení do pří-lohy zprávy. V tomto případě stačí stručný výsledek, který bude uložen do záhlaví *X-Spam-Sta-tus:*. Pro nastavení zobrazování stručného výsledku je nutné do konfiguračního souboru (/etc/spamassassin/local.cf, /etc/mail/spamassassin/local.cf) nástroje SpamAssassin přidat následující text:

```
### Šablona zobrazení výsledku clear_report_template report "_TESTSSCORES(, )_"
```

SpamAssassin obsahuje kromě jiného i Bayesův filtr, který je ve výchozí konfiguraci aktivovaný. V tomto příkladu bude filtr deaktivován, protože vyžaduje trénink, který může být jiný pro kaž-dého uživatele, a proto není vhodný pro globální filtrování protokolu SMTP:

```
### Deaktivace Bayesova filtru use_bayes 0
```

Aby se změny projevily, je nutné restartovat démona SpamAssassin (spamd).

Uživatelská nastavení a data

V některých situacích je nutné specifikovat individuální nastavení softwaru SpamAssassin, jako je například prahová hodnota pro klasifikaci spamu, povolené jazykové a znakové sady, seznamy povolených a zakázaných položek pro odesílatele atd. přímo pro jednotlivé uživatele. Uživatelé také mohou chtít individuálně využívat bayesovské filtrování, které je součástí softwaru SpamAssassin.

Poznámka

I když platí, že trénink Bayesova filtru je individuální záležitost každého uživatele, je nutné poznamenat, že Bayesův filtr v nástroji SpamAssassin není v některých případech vhod-ný. Je to znát hlavně u zpráv, do kterých spameři zahrnou velké množství náhodných slov ze slovníku nebo celé odstavce textu (například jako metadata ve zprávách ve formátu HTML).

Jak bylo zmíněno v předchozích kapitolách tohoto dokumentu, je možné vynutit, aby zpráva obsa-hovala pouze jediného příjemce. Lze toho docílit tak, že se od volajícího hostitelského systému akceptuje první příkaz RCPT TO: a další příjemci jsou odmítnuti odezvou s kódem 451. Podob-ně jako u greylistingu platí, že pokud je odesílatel legitimní agent přenosu zpráv, ví, jak tuto odez-vu vyhodnotit, a pokusí se o doručení zprávy dalším příjemcům později.

Jak akceptovat pouze jednoho příjemce v prostředí Exim

Do seznamu acl_rcpt_to se vloží za ověření adresy příjemce, ale ještě před jakýkoliv příkaz accept, který náleží k neautentizovaným doručením od vzdálených hostitelů místním uživatelům (tj. před greylisting, před kontroly signatur atd.), následující příkazy:

```
# Omezení počtu příjemců v každé příchozí zprávě na jednoho # s cílem umožnit podporu pro individuální nastavení parametrů # (např. pro SpamAssassin). # # POZNÁMKA: Každá zpráva zaslaná více uživatelům bude zpožděna
#       o 30 minut nebo více na jednoho příjemce. To může významným
#       způsobem ovlivnit interaktivní diskuse, které využívají zprávy
#       elektronické pošty a jichž se účastní jak místní, tak
#       vzdálení účastníci.
#
defer
```

message = Akceptuje se pouze jeden příjemce zprávy. condition = $recipients_count

Předání jména uživatele do nástroje SpamAssassin

V seznamu acl_data se provede úprava podmínky spam z předchozí kapitoly tak, že se nástroji SpamAssassin předá uživatelské jméno z adresy příjemce zprávy.

Volání nástroje SpamAssassin pro získání hodnot $spam_score # a $spam_report. # V závislosti na klasifikaci obsahuje proměnná $acl_m9 hodnotu # “ham” nebo “spam”. # # Předává se uživatelské jméno, které je součástí adresy příjemce, # tj. část před prvním výskytem znaku „=“ nebo „@“, převedená na malá # písmena. Zpráva by díky předchozímu omezení již neměla mít více # příjemců. # # Pokud je zpráva klasifikována jako spam, je odmítnuta. # warn
set acl_m9 = ham spam = ${lc:${extract{1}{=@}{$recipients}{$value}{mail}}} set acl_m9 = spam control = fakereject logwrite = :reject: Odmítnutý spam (score $spam_score): $spam_report

Všimněte si, že místo použití funkce *${local_part:...}* bylo uživatelské jméno získáno ručním oddělením části adresy před znakem „=“ nebo „@“. Je to díky tomu, že další znaky využijeme při kontrole signatury.

Aktivace individuálního nastavení uživatelů v nástroji SpamAssassin

Zaměříme se teď opět na SpamAssassin. Jako první změnu je možné provést odstranění nastavení parametru pro deaktivaci Bayesova filtru, který globálně jeho použití zakázal. V dalším nastavení si bude moci každý uživatel sám určit, jestli toto nastavení použije nebo ne.

Pokud se poštovní schránky na poštovním serveru mapují přímo na místní uživatelské účty s domovskou složkou, je vše hotovo. Démon SpamAssassin ve výchozí konfiguraci provádí funkci setuid() s uživatelským jménem z parametru a ukládá uživatelská data a nastavení přímo do
domovské složky uživatele. Pokud jsou poštovní schránky řešeny jinak (například je spravuje software Cyrus SASL nebo jiný poštovní server), je nutné nástroji SpamAssassin sdělit, kde nalézt konfiguraci pro každého uživa-tele a jeho soubory dat. Démon spamd také musí běžet pod účtem vybraného místního uživate-le. To vše proto, aby se nepokoušel provést funkci setuid() se jménem neexistujícího uživatele.

Vše se zajistí zadáním příslušných parametrů, předaných démonu spamd při startu:

Na systému Debian je nutné upravit nastavení OPTIONS= v souboru /etc/default/spa-massassin.
Na systému Red Hat je nutné upravit nastavení SPAMDOPTIONS= v souboru /etc/syscon-fig/spamassassin.

- U ostatních systémů je nutné příslušné parametry zjistit samostatně. Dále je možné použít

nastavení:

- -u username udává uživatele, pod kterým bude spamd spuštěn.
- -x deaktivuje konfigurační soubory v domovských složkách uživatelů.
- --virtual-config-dir=/var/lib/spamassassin/%u udává uložení individuálních nastavení jednotlivých uživatelů. Znaky „%u“ jsou nahrazeny příslušným uživatelským jménem. Démon spamd musí mít oprávnění pro vytvoření a úpravy této složky:

  # mkdir /var/lib/spamassassin# chown -R mail:mail /var/lib/spamassassin

Po provedení těchto změn je samozřejmě nutné restartovat démona spamd.

Implementace kontroly signatury adresy odesílatele

V této kapitole naleznete informace o implementaci signatury adresy odesílatele v odchozí poště a kontrolu signatury ve zprávách bez odesílatele (např. oznámení o stavu doručení). Adresa odesílatele v odchozích zprávách bude upravena takto:

odesílatel=příjemce=doména.příjemce=kontrolní_součet@doména.odesílatele

Protože tato metoda může mít vedlejší dopady (například v případě mailinglistů), je nutné ji imple-mentovat tak, aby umožnila individuální nastavení pro každého uživatele. Signatura adresy odesí-latele bude vytvořena pro odchozí zprávy pouze v případě, kdy je v domovské složce odesílate-le zprávy umístěn soubor s názvem .return-path-sign, a pouze v případě, kdy doména, do které je zpráva zasílána, je součástí tohoto souboru. Pokud soubor existuje, ale je prázdný, bude signatura vytvořena pro všechny domény.

Podobně platí, že signatura u adresy příjemce bude vyžadovaná pouze u příchozích vrácených zpráv (tzn. zpráv bez odesílatele) v případě, kdy v domovské složce příjemce existuje výše uve-dený soubor. Uživatelé mohou z této kontroly vyjmout konkrétní hostitelské systémy díky sezna-mu povolených položek.

Protože tento způsob vyžaduje spolupráci s komponentami *router* a *transport* softwaru Exim, nebude součástí výsledné verze seznamů řízení přístupu. Pokud je čtenář schopen pochopit postup uvedený v následujících kapitolách, pak určitě bude schopen přidat příslušnou sekci do seznamu řízení přístupu samostatně.

Sekce pro vytvoření signatury adresy odesílatele

Nejprve je nutné vytvořit *transport*, který se použije pro vytvoření signatury adresy odesílatele pro doručení do vzdálených domén:

remote_smtp_signed: debug_print = “T: remote_smtp_signed for $local_part@$domain” driver = smtp max_rcpt = 1 return_path =
  $sender_address_local_part=$local_part=$domain=\
                ${hash_8:${hmac{md5}{SECRET}{${lc:\ $sender_address_local_part=$local_part=

$domain}}}}\ @$sender_address_domain

Část s uživatelským jménem adresy odesílatele teď obsahuje následující části, oddělené znakem rovná se („=“):

Uživatelské jméno odesílatele.
Uživatelské jméno adresy příjemce.
Doménu z adresy příjemce.
■ Řetězec, jedinečný pro danou kombinaci odesílatele a příjemce, vygenerovaný:
Zašifrováním tří předchozích částí pomocí funkce Eximu ${hmac{md5}…} s využitím řetězce s heslem, které bylo deklarováno v hlavní části.
Vytvořením hašovaného řetězce, který sestává z 8 malých písmen s využitím funkce Eximu ${hash…}.

Poznámka

Pokud si myslíte, že vytváření haše je kanón na vrabce, musím s vámi souhlasit jen čás-tečně. V předchozí verzi dokumentu byla použita pro vytvoření poslední komponenty sig-natury funkce ${hash_8:SECRET=....}. Při použití této funkce však je možné, s přihléd-nutím k tomu, jak pracuje funkce ${hash....} v Eximu, a s využitím vzorků odchozích zpráv pro různé příjemce, zjistit, jakým způsobem je signatura vytvářená. A jak správně poznamenal Matthew Byng-Maddic <mbm@colondot.net>: Informace uvedené v tomto dokumentu si většina lidí prostě zkopíruje. No a pokud je veškeré zabezpečení součástí klíče a klíč může být zpětně odhalen, což je pravděpodobně možné při využití několika vzorků dat, spamer může začít generovat platné signatury pro libovolnou doménu – a jsme zpět na začátku. Podle mého je vhodné být tvrdý hned od začátku.

Pokud je nutné provádět autentizaci pro doručování přes vyhrazený hostitelský systém (tzv. smart-host), je možné použít i *hosts_try_auth*.

Vytvoření směrování pro doručování do vzdálených systémů

Nejprve je nutné před již existující směrovače, které obsluhují odchozí zprávy, přidat nový smě-rovač (myšleno v terminologii Eximu) zpráv. Tento směrovač využije pro doručování do vzdále-ných systémů transportní údaje z předchozí části, ale pouze v případě, pokud v domovské slož-ce odesílatele existuje soubor .return-path-sign a pokud se doména příjemce nachází v tomto souboru. Při přímém zasílání zpráv přes Internet do vzdálených cílových systémů:

```
# Vytvoření signatury adresy odesílatele pro doručení do vzdálených domén # v případě, kdy domovská složka odesílatele obsahuje soubor # ".return-path-sign" a pokud se vzdálená doména vyskytuje v tomto souboru. # Pokud soubor existuje, ale je prázdný, signatura se vytváří vždy. # dnslookup_signed: debug_print = "R: dnslookup_signed for $local_part@$domain" driver = dnslookup transport = remote_smtp_signed senders = ! : * domains = ! +local_domains : !+relay_to_domains : \
        ${if exists {/home/$sender_address_local_part/.return-path-sign}\ {/home/$sender_address_local_part/.return-path-sign}\ {!
            *}}
  no_more
```

nebo při použití vyhrazeného systému (tzv. smarthost):

```
# Vytvoření signatury adresy odesílatele pro doručení do vzdálených domén# v případě, kdy domovská složka odesílatele obsahuje soubor # ".return-path-sign" a pokud se vzdálená doména vyskytuje v tomto souboru.# Pokud soubor existuje, ale je prázdný, signatura se vytváří vždy. #smarthost_signed:
  debug_print = "R: smarthost_signed for $local_part@$domain"driver = manualroutetransport = remote_smtp_signedsenders = ! : *route_list = * smarthost.addresshost_find_failed = deferdomains = ! +local_domains : !+relay_to_domains : \
        ${if exists {/home/$sender_address_local_part/.return-path-sign}\ {/home/$sender_address_local_part/.return-path-sign}\ {!
            *}}
  no_more
```

S ohledem na ostatní použité směrovače je možné přidat další parametry (například same_doma-in_copy_routing=yes). Všimněte si, že tento směrovač se nepoužije pro zprávy, které neobsa-hují adresu odesílatele.

Poznámka

V uvedených příkladech je podmínka senders redundantní, protože soubor /home//.return-path-sign pravděpodobně neexistuje. Pro účely zpřehlednění tuto podmínku uvádíme expli-citně.

Vytvoření nového směrovače pro doručování v rámci místní domény

Dále je nutné říci Eximu, aby příchozí adresy příjemců, které jsou ve formátu uvedeném výše, doručoval do poštovní schránky identifikované částí signatury před prvním znakem rovná se („=“). Pro tyto účely je nutné ještě předtím, než doručení v rámci místní domény zajistí jiné směrovače (například směrovač *system alias*), vytvořit směrovač redirect v sekci routers v konfiguračním souboru:

```
hashed_local: debug_print = "R: hashed_local for $local_part@$domain" driver = redirect domains = +local_domains local_part_suffix = =* data = $local_part@$domain
```

Adresy příjemců, které obsahují znak rovná se, jsou překonvertovány tak, že části, které následují za znakem rovná se, jsou odříznuty. Pak se zpracují všechny ostatní směrovače.

Seznam řízení přístupu s kontrolou signatury

V poslední fázi implementace signatur je nutné nakonfigurovat Exim tak, aby zprávy doručované na platné adresy příjemců se signaturou byly akceptovány *vždy*. Ostatní zprávy s prázdnou adre-sou odesílatele budou odmítnuty pouze v případě, kdy příjemce využívá stejné schéma. Ani v jed-nom případě by se neměl provádět greylisting.

Následující zdrojový kód by měl být umístěn do seznamu acl_rcpt_to před kontroly SPF grey-listing anebo poslední příkaz accept:

```
# Akceptuje adresy příjemců v případě, kdy obsahují vlastní signaturu.
# To znamená, že aktuální zpráva je odpově (oznámení o stavu doručení,
# ověření příjemce zpětným voláním) na zprávu, která pochází od nás.

#
accept
domains         = +local_domains
condition       = ${if and {{match{${lc:$local_part}}{^(.*)=(.*)}}\
                  {eq{${hash_8:${hmac{md5}{SECRET}{$1}}}}{$2}}}\

                  {true}{false}}
```

# V opačném případě je to vrácen zpráva (tj. pokud neobsahuje adresu # odesílatele), ale pokud příjemce vyžaduje kontrolu signatury, dojde # k odmítnut. # deny
message = Adresa neobsahuje platnou signaturu \ z výchoz domény.\n\ Odpovídáte na
podvrženou adresu odesílatele.
log_message = falešn vrácen zpráva.senders = : postmaster@*domains = +local_domainsset
acl_m9 = /home/${extract{1}{=}{${lc:$local_part}}}\
/.return-path-sign condition = ${if exists {$acl_m9}{true}}

Při zasílání zpráv hostitelským systémům, které provádějí ověřování adres odesílatele zpětným voláním v záhlaví zprávy, například v záhlaví *From:* odchozí zprávy, může vzniknout problém. Příkaz deny v takovém případě určitě vyhodnotí takový pokus o ověření negativně. Proto je vhod-né poslední příkaz deny změnit na příkaz warn, uložit informaci o odmítnutí do proměnné $acl_m0 a provést odmítnutí až po příkazu DATA takto:

```
# V opačném případě je to vrácená zpráva (tj. pokud neobsahuje # odesílatele), ale pokud příjemce vyžaduje kontrolu signatury, dojde # k uložení
zprávy o odmítnutí do proměnné $acl_m0 a zprávy pro soubor # protokolu do proměnné $acl_m1. Tyto zprávy se využijí později při # vlastním
odmítnutí zprávy. V mezičase to znamená, že je nutné # odesílatele dále brzdit. # warn
senders = : postmaster@*domains = +local_domainsset acl_m9 = /home/${extract{1}{=}{${lc:$local_part}}}/\
.return-path-sign condition = ${if exists {$acl_m9}{true}}
    set acl_m0 = Adresa příjemce <$local_part@$domain> neobsahuje \
                 platnou signaturu z výchozí domény.\n\
                 Odpovídáte na podvrženou adresu odesílatele.
    set acl_m1 = falešná vrácená zpráva pro <$local_part@$domain>.
```

Pokud příjemce zvolí používání signatur adresy odesílatele v ochozích zprávách, může chtít nakonfigurovat výjimky z používání signatur v příchozí poště tak, že vybrané hostitelské systémy nebudou muset poskytovat signaturu v příchozích zprávách, a to ani v případě, kdy zpráva neob-sahuje adresu odesílatele. Tyto výjimky se mohou uplatnit u některých vybraných konferenčních serverů. Více informací naleznete v části Signatury adresy odesílatele.

Akceptace vrácených zpráv pouze pro reálně existující uživatele

Jak bylo zmíněno v části o blokování zavlečeného spamu, existuje slabé místo, které brání v zachytávání falešných zpráv s oznámeními o stavu doručení, zasílaných na systémové uživatele a aliasy (např. *postmaster*). V této kapitole naleznete dva způsoby, jak zajistit, že vrácené zprávy jsou akceptovány pouze pro uživatele, kteří generují odchozí poštu.

Kontrola poštovní schránky příjemce

Tato kontrola se provádí v seznamu acl_rcpt_to. Kontroluje se, zda adresa příjemce odpovídá nějaké místní poštovní schránce:

```
# Odmítnutí zpráv pro uživatele, kteří nemají vlastní poštovní schránku
# (tj. postmaster, webmaster atd.), v případě, kdy zpráva neobsahuje
# adresu odesílatele. Tito uživatelé neodesílají žádné zprávy,
# takže by neměli dostávat ani žádné vrácené zprávy.
#
```

```
deny
   message = Tato adresa nezasílá poštovní zprávy. \
                                          Odpovídáte na falešnou adresu odesílatele.
   log_message = falešná vrácená zpráva pro <$local_part@$domain>
   senders = : postmaster@*
   domains = +local_domains
   !mailbox check
```

Kontrola `mailbox check` závisí na tom, jakým způsobem se provádí doručování zpráv (i v tomto případě se odděluje část před prvním znakem rovná se („=“) v adrese příjemce, aby se ošetřil pří-pad signatury adresy).

- Pokud jsou poštovní schránky mapovány na místní uživatelské účty na serveru, je možné kontrolovat, jestli je možné mapovat jméno příjemce na identifikátor uživatele, který odpo-vídá skutečným uživatelům v systému, tj. v rozsahu 500–60 000:

```
set acl_m9 = ${extract{1}{=}{${lc:$local_part}}}
set acl_m9 = ${extract{2}{:}{${lookup passwd{$acl_m9}{$value}}}{0}}
condition = ${if and{{>={$acl_m9}{500}}{<${acl_m9}{60000}}}{true}}
```

- Pokud jsou zprávy doručovány pomocí softwaru Cyrus s využitím protokolu IMAP, je možné pro kontrolu existence dané poštovní schránky použít nástroj mbpath. Pak je samozřejmě nutné zajistit, aby uživatel, pod kterým je Exim spuštěn, měl oprávnění pro kontrolu poštovních schránek (například je možné tohoto uživatele přidat do skupiny *cyrus*: #adduser exim4 cyrus).

```
set acl_m9 = ${extract{1}{=}{${lc:$local_part}}}
condition = ${run {/usr/sbin/mbpath -q -s user.$acl_m9} {true}}
```

- Pokud jsou všechny zprávy směrovány na jiný počítač, který provádí jejich doručování, je možné provádět ověření adresy příjemce zpětným voláním a nechat tento jiný počítač roz-hodnout, jestli zprávu přijme. Při provádění zpětného volání je nutné ponechat původní adresu odesílatele nezměněnou:

```
verify = recipient/callout=use_sender
```

V případě místně doručovaných zpráv tato kontrola poštovních schránek duplikuje funkce prová-děné směrovačem, a protože je specifická pro mechanismus doručování zpráv na poštovním ser-veru z našeho příkladu, může perfekcionistům z řad čtenářů připadat trochu nečitelná.

Kontrola prázdné adresy odesílatele ve směrovači pro zpracování aliasů

V konfiguraci pravděpodobně máte obsažen směrovač s názvem `system_aliases` pro přesměro-vání zpráv pro uživatele typu *postmaster* nebo *mailer-demon*. Tyto aliasy se obvykle nepoužívají jako adresy odesílatele v ochozích zprávách. Proto je možné zajistit, aby příchozí oznámení o stavu doručení nebyla takovým směrovačem směrována přidáním následující podmínky:

```
!senders = : postmaster@*
```

Příklad směrovače pro aliasy pak může vypadat například takto:

```
system_aliases:
   driver = redirect
   domains = +local_domains
   !senders = : postmaster@*
   allow_fail
   allow_defer
   data = ${lookup{$local_part}lsearch{/etc/aliases}}
   user = mail
   group = mail
   file_transport = address_file
   pipe_transport = address_pipe
```

V tomto okamžiku jsou blokovány vrácené zprávy na *některé* systémové aliasy, nicméně ostatní aliasy víceméně odpovídají existujícím systémovým uživatelům (např. „root“, „daemon“ atd.). Pokud jsou místně doručované zprávy doručovány přes ovladač *accept* a pro ověření adresy pří-jemce se využívá kontrola *check_local_user*, jsou v tomto okamžiku zprávy směrovány přímo na tyto systémové účty.

Řešení tohoto problému spočívá v přidání další dodatečné podmínky do směrovače, který řídí doručování místních zpráv (tj. např. *local_user*), kterou se zkontroluje, že příjemce existuje a navíc je to „normální“ uživatel. Kontrolu je možné například provést porovnáním identifikátoru uživatele, který by měl ležet v rozsahu 500–60 000:

```
condition = ${if and {{>={$local_user_uid}{500}}\
                      {<{$local_user_uid}{60000}}}\
                      {true}}
```

Příklad směrovače pro doručování místních zpráv tedy může vypadat takto:

```
local_user:
   driver = accept
   domains = +local_domains
   check_local_user
   condition = ${if and {{>={$local_user_uid}{500}}\
                                                {<{$local_user_uid}{60000}}}\
                                               {true}}
   transport = transport
```

Při implementaci této metody je nutné mít na paměti, že odmítnutí falešné vrácené zprávy bude realizováno odezvou s kódem 550 Unknown User, tj. stejnou odezvou, která se používá pro odmítnutí zprávy v případě neexistujícího příjemce.

Výjimky pro přeposílané zprávy

Po přidání všech uvedených kontrol do transakcí protokolu SMTP určitě zjistíte nepřímé genero-vání zavlečeného spamu jako výsledek odmítnutí zpráv přeposílaných z důvěryhodných zdrojů, jako jsou například mailinglisty nebo poštovní účty v jiných hostitelských systémech. Takové hos-titelské systémy je proto nutné přidat do seznamu povolených položek a vyjmout je z procesu odmítnutí – minimálně těch odmítnutí, která vzniknou v důsledku antivirové nebo antispamové kontroly.

V tomto příkladu se bude provádět při zpracování příkazu RCPT TO: kontrola obsahu dvou sou-borů:

Globální seznam povolených položek /etc/mail/whitelist-hosts, který obsahuje záložní poštovní servery a ostatní povolené odesílatele
Individuální seznam uživatelských povolených položek /home/user/.forwarders, který obsahuje hostitelské systémy, ze kterých může daný uživatel přijímat přeposílané zprávy (tzn. mailinglisty nebo poštovní servery odchozích zpráv z jiných domén atd.)

Pokud uživatelé poštovního serveru nemají místní uživatelské účty a domovské složky, je nutné cesty k souborům a samotné vyhledávání přizpůsobit konkrétnímu systému (například vyhledá-vání v databázi nebo dotazy do adresáře LDAP). Pokud je daný hostitelský systém odesílatele nale-zen v některém z těchto seznamů, uloží se do proměnné $acl_m0 hodnota „accept“ a obsah pro-měnné $acl_m1 se smaže tak, jak je to popsáno v předchozí části. To je indikátor, který udává, že zpráva by neměla být v průběhu zpracování dalších příkazů transakce protokolu SMTP odmít-nuta.

Do seznamu acl_rcpt_to se hned za ověření adresy příjemce, ale před příkazy accept, které se týkají kontrol neautentizovných doručení zpráv ze vzdálených hostitelských systémů místním uži-vatelům (tj. před greylisting, kontroly signatur atd.), vloží následující kód:

```
   # Akceptuje zprávu v případě, kdy je hostitelský systém odesílatele
   # uveden v globálním seznamu povolených položek. Dočasně nastaví
   # proměnnou $acl_m9 tak, aby odkazovala na tento soubor.
   # Pokud je hostitelský systém v seznamu nalezen, nastaví proměnnou
   # $acl_m0 a smaže hodnotu proměnné $acl_m1, která zabrání následnému
   # pozdějšímu odmítnutí zprávy.
   #
   accept
      set acl_m9 = /etc/mail/whitelist-hosts
      hosts = ${if exists {$acl_m9}{$acl_m9}}
      set acl_m0 = accept
      set acl_m1 =


   # Akceptuje zprávu v případě, kdy je hostitelský systém odesílatele # uveden v souboru „.forwarders“ v domovské složce příjemce zprávy.#
   Dočasně nastaví proměnnou $acl_m9 tak, aby odkazovala na tento soubor. # Pokud je hostitelský systém v souboru nalezen, nastaví
   proměnnou# $acl_m0 a smaže hodnotu proměnné $acl_m1, která zabrání následnému # pozdějšímu odmítnutí zprávy.
   #
   acceptdomains = +local_domainsset acl_m9 = /home/${extract{1}{=}{${lc:$local_part}}}/.forwardershosts = ${if exists {$acl_m9}
      {$acl_m9}} set acl_m0 = acceptset acl_m1 =
```

Kontrola obsahu proměnné $acl_m0 se provádí na více místech seznamu acl_data a zabrání se tím případnému odmítnutí legitimní zprávy. Podobným způsobem je například možné zabránit odmít-nutí zpráv od hostitelských systémů ze seznamu povolených položek v důsledku chybějících záhla-ví zprávy dle definice z dokumentu RFC 2822 takto:

```
   denymessage = Zpráva nevyhovuje standardu dle RFC 2822log_message = chybějící záhlaví!hosts = +relay_from_hosts!senders = :
      postmaster@*condition = ${if !eq {$acl_m0}{accept}{true}}condition = ${if or {{!def:h_Message-ID:}\
                                          {!def:h_Date:}\ {!def:h_Subject:}} {true}{false}}
```

Odpovídající kontroly jsou součástí konečné verze seznamů řízení přístupu v další kapitole.

Konečná verze seznamů řízení přístupu

Tak pojďme na to. Bylo to dlouhé čtení, každopádně gratuluji všem, kteří to zvládli až sem. Následující seznamy řízení přístupu zahrnují všechny kontroly, které byly popsány v této imple-mentaci. Některé však byly zakomentovány z těchto důvodů:

Greylisting. Greylisting vyžaduje buď instalaci dodatečného softwaru nebo poměrně složitou konfiguraci dodatečných seznamů řízení přístupů a definicí v konfiguračním souboru Eximu. I přesto však tuto metodu doporučuji.
Antivirový program. Bohužel neexistuje jediný antivirový program, který používají všichni tak, jak je tomu u antispamového programu v podobě softwaru SpamAssassin. Implemen-tace antivirových kontrol je však s pomocí dokumentace, která je součástí doplňku *Exis-can-ACL*, poměrně snadná.
Individuální uživatelská nastavení pro SpamAssassin. Pro mnohé je to implementačně neakceptovatelné, protože zahrnuje zpožďování zpráv pro všechny příjemce zprávy s výjimkou prvního.
Signatury adresy odesílatele. Existují omezení pro uživatele, kteří se připojují v různých sítích. Kromě konfigurace seznamů řízení přístupu také vyžaduje i konfiguraci směrovačů a komponent pro transport zpráv.
Akceptace vrácených zpráv pouze pro reálně existující uživatele. Existuje více způsobů

implementace, které jsou závislé na způsobu doručování zpráv. Následují jednotlivé konečné verze seznamů řízení přístupu.

Seznam acl_connect

```
# Tento seznam se použije na začátku transakce při příjmu příchozího
# připojení. Testy se provádí, aby se zjistilo, jestli příchozí připojení
# přijmout nebo odmítnout.
acl_connect:
# Zaznamená aktuální čas pro výpočet doby trvání zpoždění. warn
set acl_m2 = $tod_epoch


# Akceptování zprávy přijaté z lokální služby SMTP (tj. ne přes TCP/IP)
# Kontrola se provádí pro prázdnou položku odesílajícího host. systému.
# Akceptuje zprávy přijaté z hostitelských systémů, pro které se provád
í# přeposílání (relay) zpráv.accept
hosts = : +relay_from_hosts
# Pokud se hostitelský systém nachází v některém seznamu zakázaných
# položek systému DNS, uloží se varování do proměnné $acl_c1. Toto
# varování bude později přidáno do záhlaví zprávy. V mezičase hodnota
# této proměnné indikuje, že je nutné pokračovat v brzdění odesílatele
# zprávy.
#
warn
!hosts = ${if exists {/etc/mail/whitelist-hosts} \
                {/etc/mail/whitelist-hosts}}
                dnslists = list.dsbl.org : \
                dnsbl.sorbs.net : \
                dnsbl.njabl.org : \
                bl.spamcop.net : \
                dsn.rfc-ignorant.org : \
                sbl-xbl.spamhaus.org : \
                l1.spews.dnsbl.sorbs.net
set acl_c1 = X-DNSbl-Warning: \
                $sender_host_address je v seznamu $dnslist_domain\
                ${if def:dnslist_text { ($dnslist_text)}}
# Pokud selže reverzní vyhledávání v systému DNS (tj. neexistuje
# položka rDNS nebo dopřední vyhledávání názvu nevrátí adresu IP,
# která je shodná s původní adresou IP), pak se vygeneruje varování
# do proměnn $acl_c1. Toto varován bude později uloženo do záhlaví
# zprávy.
warn
condition = ${if !def:acl_c1 {true}{false}} !
verify = reverse_host_lookup
set acl_m9 = Reverzn vyhledán DNS selhalo pro $sender_host_address
set acl_c1 = X-DNS-Warning: $acl_m9
```

```
# Akceptuje připojení. Pokud je v proměnné $acl_c1 uloženo nějaké
# varování, bude transakce protokolu SMTP zpožděna o 20 sekund. accept
set acl_m2 = ${if def:acl_c1 {${eval:20 + $acl_m2 - $tod_epoch}}{0}}
delay = ${if >{$acl_m2}{0}{$acl_m2}{0}}s
Seznam acl_helo
# Tento seznam se použije pro kontrolu příkazu HELO nebo EHLO
# v příchozí transakci protokolu SMTP. Testy se provádí, aby se zjistilo,
# jestli transakci při zpracování příkazu HELO nebo EHLO přijmout nebo
# odmítnout.
acl_helo:
# Zaznamená aktuální čas pro výpočet doby trvání zpoždění. warn
set acl_m2 = $tod_epoch
# Akceptování zprávy přijaté z lokální služby SMTP (tj. ne přes TCP/IP)
# Kontrola se provádí pro prázdnou položku odesílajícího host. systému.
# Akceptuje zprávy přijaté z hostitelských systémů, pro které se provádí
# přeposílání (relay) zpráv.
#accept
hosts = : +relay_from_hosts
# Pokud vzdálený hostitelský systém uvede svou adresu IP, připraví se
# zpráva o odmítnutí do proměnné $acl_c0 a zpráva pro soubor protokolu
# do proměnné $acl_c1. Později budou použity v příkazu „deny“.
# V mezičase hodnota této proměnné indikuje, že je nutné pokračovat
# v brzdění odesílatele zprávy.
# warn
condition = ${if isip {$sender_helo_name}{true}{false}}
set acl_c0 = Zpráva byla doručena ratwarem
set acl_c1 = vzdálený hostitel použil adresu IP v příkazu HELO/EHLO
# Podobně v případě, kdy hostitelský systém použije název naší
# vlastní domény.
#
warn
condition = ${if match_domain{$sender_helo_name}\
                {$primary_hostname:+local_domains:+relay_to_domains}\
                {true}{false}}
set acl_c0 = Zpráva byla doručena ratwarem
set acl_c1 = vzdálený systém použil naši doménu v příkazu HELO/EHLO

# Pokud kontrola příkazu HELO selže, uloží se varování do proměnné
# acl_c1. Toto varován se později přid do záhlav zprávy.
# V mezičase hodnota těchto proměnných indikuje, že je nutné pokračovat
# v brzdění odesílatele zprávy.
# warn
condition = ${if !def:acl_c1 {true}{false}}!
verify = heloset
acl_c1 = X-HELO-Warning: Vzdálený host $sender_host_address \
                ${if def:sender_host_name {($sender_host_name) }}\
                se vydáv za $sender_helo_name
log_message = vzdálený host použil neověřitelný příkaz HELO/EHLO.
# Akceptuje připojení. Pokud je v proměnn $acl_c1 uloženo nějaké
# varování, bude transakce protokolu SMTP zpožděna o 20 sekund.
accept
set acl_m2 = ${if def:acl_c1 {${eval:20 + $acl_m2 - $tod_epoch}}{0}}
delay = ${if >{$acl_m2}{0}{$acl_m2}{0}}s
Seznam acl_mail_from
# Tento seznam řízení přístupu se použije pro příkaz MAIL FROM: v příchozí
# transakci protokolu SMTP. Testy se provádějí, aby se zjistilo, jestli
# zprávu přijmout nebo odmítnut na základě adresy odesílatele.
#
acl_mail_from:
# Zaznamená aktuální čas pro výpočet doby trvání zpoždění.
warn
```

```
set acl_m2 = $tod_epoch
# Akceptování zprávy přijaté z lokální služby SMTP (tj. ne přes TCP/IP)
# Kontrola se provádí pro prázdnou položku odesílajícího host. systému.
# Akceptuje zprávy přijaté z hostitelských systémů, pro které se provádí
# přeposílání (relay) zpráv.
# # Ověření odesílatele se zde neprovádí, protože klientské poštovní
# aplikace jsou ve většině případů jednoduché a neumějí správně zpracovat
# chybové odezvy protokolu SMTP.
# accept
hosts = : +relay_from_hosts
# Akceptování zprávy, která je doručována autentizovaným připojením
# z libovolného hostitelského systému. Takové zprávy pocházejí většinou
# přímo od klientských poštovních aplikací.
# accept
authenticated = *
# Proměnné $acl_c0 a $acl_c1 obsahují zprávu o odmítnutí nebo varování,
# které je nutné použít při každém pokusu o doručení v rámci aktuální
# transakce protokolu SMTP. Jejich hodnotu je nutné přiřadit
# odpovídajícím proměnným $acl_m{0,1}, které jsou platné po dobu
# zpracování zprávy, a přidat varování z proměnné $acl_m1 do záhlaví
# zprávy. (V případě odmítnutí zprávy proměnná $acl_m1 obsahuje zprávu
# pro soubor protokolu, nicméně záhlaví bude zrušeno současně
# s odmítnutou zprávou).
#
warnset acl_m0 = $acl_c0
set acl_m1 = $acl_c1
message = $acl_c1
# Pokud odesílatel nepoužil příkaz HELO/EHLO, do proměnné $acl_m0 se
# uloží zpráva o odmítnutí a zpráva pro soubor protokolu do proměnné
# $acl_m1. Tyto zprávy budou později použity v příkazu „deny“.
# V mezičase hodnota této proměnné indikuje, že je nutné pokračovat
# v brzdění odesílatele zprávy.
# warn
condition = ${if def:sender_helo_name {0}{1}}
set acl_m0 = Zpráva byla doručena ratwaremset
acl_m1 = Vzdálený hostitel nepoužil příkaz HELO/EHLO.
# Pokud není možné ověřit adresu odesílatele, do proměnné $acl_m1 se
# uloží varování, které se přidá do záhlaví zprávy. Přítomnost této
# zprávy udává, že je nutné pokračovat v brzdění odesílatele zprávy.
# # Dle uvážení je možné vynechat možnost „callout”. Pokud jsou odchozí
# zprávy zasílány s využitím vyhrazeného hostitelského systému (tzv.
# smarthost), nedá ověření vzdáleným voláním žádné užitečné výsledky.
# warn
condition = ${if !def:acl_m1 {true}{false}}!
verify = sender/calloutset
acl_m1 = Neplatný odesílatel <$sender_address>message = X-Sender-Verify-Failed: $acl_m1
log_message = $acl_m1
# Akceptuje odesílatele. Pokud je v proměnné $acl_c1 uloženo nějaké
# varování, bude transakce protokolu SMTP zpožděna o 20 sekund.accept
set acl_m2 = ${if def:acl_c1 {${eval:20 + $acl_m2 - $tod_epoch}}{0}}
delay = ${if >{$acl_m2}{0}{$acl_m2}{0}}s
```

Seznam acl_rcpt_to

```
# Tento seznam řízení přístupu se použije pro příkaz RCPT v příchozí
# transakci protokolu SMTP. Testy se provádějí, aby se zjistilo, jestli
# zprávu přijmout nebo odmítnut na základě adresy příjemce.
#
acl_rcpt_to:
# Akceptování zprávy přijaté z lokální služby SMTP (tj. ne přes TCP/IP)
# Kontrola se provádí pro prázdnou položku odesílajícího host. systému.
# Akceptuje zprávy přijaté z hostitelských systémů, pro které se provádí
```

```
# přeposílání (relay) zpráv.
#
# Ověření odesílatele se zde neprovádí, protože klientské poštovní
# aplikace jsou ve většině případů jednoduché a neumějí správně zpracovat
# chybové odezvy protokolu SMTP. accept
hosts = : +relay_from_hosts
# Akceptování zprávy, která je doručována autentizovaným připojením
# z libovolného hostitelského systému. Takové zprávy pocházejí většinou
# přímo od klientských poštovních aplikací.
# accept
authenticated = *
# Odmítnutí zprávy se provede, pokud jméno v adrese obsahuje znaky @
# nebo % nebo / nebo | nebo !. Tyto znaky se ve jméně vyskytují velmi
# zřídka, ale občas je zkouší použít uživatelé, kteří se snaží
# obejít omezení při přeposílání zpráv.
# # Odmítnutí zprávy se provede i v případě, kdy jméno začíná tečkou.
# Prázdné části adresy nejsou podle striktního výkladu dokumentu
# RFC 2822 povoleny, ale Exim je umožňuje využít, protože
# se obvykle používají. Pokud adresa začíná tečkou, může to způsobit
# problém v případě, kdy se jméno používá jako název souboru (například
# pro mailinglist).
# deny
local_parts = ^.*[@%!/|] : ^\\.
# Odmítnutí zprávy v případě, kdy proměnná $acl_m0 obsahuje zprávu
# o odmítnutí. Kromě toho se provede upoždění transakce protokolu SMTP
# o 20 sekund.
# deny
message = $acl_m0
log_message = $acl_m1
condition = ${if and {{def:acl_m0}{def:acl_m1}} {true}}delay = 20s
# Pokud je adresa příjemce z jiné domény, než pro kterou se pošta
# zpracovává, provede se zpoždění transakce a odmítnutí.
# deny
message = přeposílání (relay) zpráv není dovoleno!
domains = +local_domains : +relay_to_domainsdelay = 20s
# Pokud je adresa příjemce z místní domény nebo z domény, pro kterou se
# provádí přeposílání pošty, ale je neplatná, provede se zpoždění
# transakce a odmítnutí.
#
deny
message = neznámý příjemce !
verify = recipient/call
out=20s,defer_ok,use_sender
delay = ${if def:sender_address {1m}{0s}}
# Ukončení připojení se provede v případě, kdy je adresa odesílatele
# prázdná, ale zpráva má více adres příjemců. Legitimní oznámení
# o stavu doručen není nikdy zasláno na více adres.
#
drop
message = Legitimní oznámení o stavu doručení nejsou zaslána \
          více příjemcům.
senders = : postmaster@*
condition = $recipients_count
delay = 5m
# Omezení počtu příjemců v každý příchozí zprávě na jednoho
# s cílem umožnit podporu pro individuální nastavení parametrů
# (např. pro SpamAssassin).
#
# POZNÁMKA: Každá zpráva zaslaná více uživatelům bude zpožděna
# o 30 minut nebo více na jednoho příjemce. To může významným
# způsobem ovlivnit interaktivní diskuse, které využívaj zprávy
# elektronické pošty a jichž se účastní jak místní, tak
# vzdálení účastníci. Proto je tato část zakomentovaná.
```

```
#
defer message = Akceptuje se pouze jeden příjemce zprávy.
condition = $recipients_count
# Akceptuje zprávu v případě, kdy je hostitelský systém odesílatele
# uveden v souboru „.forwarders“ v domovské složce příjemce zprávy.
# Dočasně nastaví proměnnou $acl_m9 tak, aby odkazovala na tento soubor.
# Pokud je hostitelský systém v souboru nalezen, nastaví proměnnou
# $acl_m0 a smaže hodnotu proměnní $acl_m1, která zabrání následnému
# pozdějšímu odmítnutí zprávy.
#
accept domains = +local_domains
set acl_m9 = /home/${extract{1}{=}{${lc:$local_part}}}/.forwarders
hosts = ${if exists {$acl_m9}{$acl_m9}}
set acl_m0 = accept
set acl_m1 =
# Akceptuje zprávu v případě, kdy je hostitelský systém odesílatele
# uveden v globálním seznamu povolených položek. Dočasně nastaví
# proměnnou $acl_m9 tak, aby odkazovala na tento soubor.
# Pokud je hostitelský systém v seznamu nalezen, nastaví proměnnou
# $acl_m0 a smaže hodnotu proměnný $acl_m1, který zabrání následnému
# pozdějšímu odmítnutí zprávy.
# accept
set acl_m9 = /etc/mail/whitelist-hosts
hosts = ${if exists {$acl_m9}{$acl_m9}}
set acl_m0 = accept
set acl_m1 =
# ------------------------------------------------------------------------
# Kontrola signatury adresy odesílatele.
# Tato část je zakomentovaná, protože vyžaduje další konfiguraci
# v částech ‘transports’ a ‘routers’.
## Akceptuje adresy příjemců v případě, kdy obsahují vlastní signaturu.
# To znamen, že aktuální zpráva je odpověď (oznámení o stavu doručení,
# ověření příjemce zpětným voláním) na zprávu, která pochází od nás.
### accept
# domains = +local_domains
# condition = ${if and {{match{${lc:$local_part}}{^(.*)=(.*)}}\
# {eq{${hash_8:${hmac{md5}{SECRET}{$1}}}}{$2}}}\
# {true}{false}}
# V opačném případě je to vrácená zpráva (tj. pokud neobsahuje
# odesílatele, ale pokud příjemce vyžaduje kontrolu signatury, dojde
# k odmítnutí.
## deny
# message = Adresa neobsahuje platnou signaturu \
# z výchozí domény.\n\
# Odpovídáte na podvrženou adresu odesílatele.
# log_message = falešná vrácená zpráva.
# senders = : postmaster@*
# domains = +local_domains
# set acl_m9 = /home/${extract{1}{=}{${lc:$local_part}}}\
# /.return-path-sign
# condition = ${if exists {$acl_m9}{true}}
# ------------------------------------------------------------------------
# ------------------------------------------------------------------------
# Odmítnutí zpráv pro uživatele, kteří nemají vlastní poštovní schránku
# (tj. postmaster, webmaster atd.), v případě, kdy zpráva neobsahuje
# adresu odesílatele. Tito uživatelé neodesílají žádné zprávy,
# takže by neměli dostávat ani žádné vrácené zprávy.
# Poznámka: Tato kontrola je zakomentovaná, protože závis na
# způsobu doručování místních zpráv. Pokud chcete tuto kontrolu
# použít, je nutné okomentovat pouze jednu z dále uvedených podmínek.
#deny
```

```
# message = Tato adresa nezasílá poštovní zprávy. \
# Odpovídáte na falešnou adresu odesílatele.
# log_message = falešně vrácená zpráva pro <$local_part@$domain>
# senders = : postmaster@*
# domains = +local_domains
# set acl_m9 = ${extract{1}{=}{${lc:$local_part}}}
## --- Pokud mají příjemci místní uživatelsk účty, odkomentovat 2 řádky:
# set acl_m9 = ${extract{2}{:}{${lookup passwd {$acl_m9}{$value}}}{0}}
# !condition = ${if and {{>={$acl_m9}{500}} {<${acl_m9}{60000}}}{true}}
# # --- Pokud se doručuje s využitím softwaru Cyrus, odkomentovat řádek:
# condition = ${run {/usr/sbin/mbpath -q -s user.$acl_m9} {true}}
# ------------------------------------------------------------------------
# Dotaz na informace SPF pro doménu odesílatele zprávy.
# Podle výsledku se rozhodne, jestli je hostitelský systém odesílatele
# zprávy autorizovaný pro doručen zprávy.
# Pokud ne, zpráva je odmítnuta.
#
deny

message = [SPF] $sender_host_address nem umožněno zasílat zprávy \
          z domény $sender_address_domain
log_message = kontrola SPF selhala
spf = fail
# Přidán záhlav SPF-Received: do zprávy warn message = $spf_received
# Pro získán stavu pro daný triplet (hostitel, odesílatel, příjemce) se
# využije démon „greylistd“.
# Před použitím dalšího kroku je nutné nainstalovat démona „greylistd“.
# Viz: http://packages.debian.org/unstable/main/greylistd
#
# Greylisting se neprovádí pro zprávy bez odesílatele, protože ověření
# odesílatele zpětným voláním by nefungovalo (a pravděpodobně by nebylo
# možné zaslat zprávu systému, který toto ověřování provádí).
#
# defer
# message = $sender_host_address ještě není autorizován pro
# doručení\
# zprávy z adresy <$sender_address> na adresu\
# <$local_part@$domain>. Prosím zkuste později.
# log_message = blokováno greylistingem.
# domains = +local_domains : +relay_to_domains
# !senders = : postmaster@*
# set acl_m9 = $sender_host_address $sender_address
# $local_part@$domain
# set acl_m9=${readsocket{/var/run/greylistd/socket}{$acl_m9}{5s}{}{}}
# condition = ${if eq {$acl_m9}{grey}{true}{false}}
# delay = 20s
# ------------------------------------------------------------------------
# Akceptuje příjemce. accept
```

Seznam acl_data

```
# Tento seznam řízení přístupu se využívá pro kontrolu dat obsahu zprávy.
# Testy se provádějí, aby se zjistilo, jestli zprávu přijmout nebo
# odmítnout na základě adresy příjemce.
acl_data: # Zaznamenání záhlaví warn
logwrite = Subject: $h_Subject:
# Přidání záhlaví Message-ID do zprávy zaslané klienty z vlastní domény
# v případě, kdy záhlaví ve zprávě chybí. warn
condition = ${if !def:h_Message-ID: {1}}
hosts = +relay_from_hosts
message = Message-ID: <E$message_id@$primary_hostname>
# Akceptování zprávy přijaté z lokální služby SMTP (tj. ne přes TCP/IP)
```

```
# Kontrola se provádí pro prázdnou položku odesílajícího host. systému.
# Akceptuje zprávy přijaté z hostitelských systémů, pro které se provádí
# přeposílání (relay) zpráv.
#accept
hosts = : +relay_from_hosts
# Akceptování zprávy, která je doručována autentizovaným připojením
# z libovolného hostitelského systému.
#
accept

authenticated = *
# Odmítnutí zprávy v případě, kdy proměnná $acl_m0 obsahuje zprávu
# o odmítnutí. Kromě toho se provede upoždění transakce protokolu SMTP
# o 20 sekund.
# deny
message = $acl_m0
log_message = $acl_m1
condition = ${if and {{def:acl_m0}{def:acl_m1}} {true}{false}}
delay = 20s
# Vynucení kontroly omezení velikosti zprávy
#
deny
message = Velikost zprávy $message_size je větší než limit \
          MESSAGE_SIZE_LIMIT
condition = ${if >{$message_size}{MESSAGE_SIZE_LIMIT}{yes}{no}}
# Odmítnutí zprávy v případě, kdy není korektní syntaxe záhlaví.
#

deny message = Zpráva nevyhovuje specifikaci standardu z RFC 2822
log_message = chyba při syntaktické kontrole záhlaví zprávy !
verify = header_syntax
# Následující řádky odkomentujte pro odmítnutí zpráv bez záhlaví
# Message-ID:, Date: nebo Subject:.
# Někteří specializovaní agenti přenosu zpráv (některé mailinglistové
# servery) automaticky negenerují záhlaví Message-ID pro vrácené zprávy.# Proto je přidaná kontrola na neprázdnost adresy odesílatele.## deny#
message = Zpráva nevyhovuje specifikaci standardu z RFC 2822# log_message = chybějící záhlaví zprávy# !hosts = +relay_from_hosts# !senders
= : postmaster@*# condition = ${if or {{!def:h_Message-ID:}\# {!def:h_Date:}\# {!def:h_Subject:}} {true}{false}}
# Varování v případě, kdy není nalezena platná adresa odesílatele ani

# v jednom ze záhlaví „Sender:", „Reply-To:" nebo „From:".

#
warn message = X-Sender-Verify-Failed: Nenalezena platná adresa\ odesílatele
log_message = Nenalezen platný odesílatel !
verify = header_sender
# ------------------------------------------------------------------------
# Greylisting pro zprávy bez adresy odesílatele
# Pro tyto zprávy se neprovádí greylisting po příkazu RCPT TO:, protože
# by to kolidovalo se vzdálenými systémy, které provádějí ověřování
# odesílatele zpětným voláním.
# Protože je adresa odesílatele prázdná, nepoužívá se.
#
# Před použitím dalšího kódu je nutné nainstalovat démona „greylistd".
# Viz: http://packages.debian.org/unstable/main/greylistd
##defer
# message = $sender_host_address ještě není autorizován pro
# doručení\
# zprávy z adresy <$sender_address> na adresu\
# <$local_part@$domain>. Prosím zkuste později.
# log_message = blokováno greylistingem.
# senders = : postmaster@*
```

```
# condition = ${if !eq {$acl_m0}{accept}{true}}
# set acl_m9 = $sender_host_address $recipients
# set acl_m9 =${readsocket{/var/run/greylistd/socket}{$acl_m9}{5s}{}{}}
# condition = ${if eq {$acl_m9}{grey}{true}{false}}
# delay = 20s
# ---------------------------------------------------------------------

# --- ZAČÁTEK KONFIGURACE EXISCAN --

# Odmítnutí zpráv s chybami kódování MIME.
# deny
message = Chyba kódování MIME ($demime_reason)demime = *condition = ${if >{$demime_errorlevel}{2}{1}{0}}

# Rozbalení kontejnerů v kódování MIME a odmítnutí souborů s příponami,
# které využívají viry.
# Zde se provádí další volání příkazu demime, které ovšem vrátí výsledky
# uložené ve vyrovnávací paměti.
# Seznam přípon nemusí být úplný.
#
deny
message = Zde neakceptujeme přílohy s příponou „.$found_extension".
demime = bat:btm:cmd:com:cpl:dll:exe:lnk:msi:pif:prf:reg:scr:vbs:url
# Zprávy větší než MESSAGE_SIZE_SPAM_MAX jsou přijaty bez antivirové
# a antispamové kontroly accept
condition = ${if >{$message_size}{MESSAGE_SIZE_SPAM_MAX}  {true}}


logwrite = :main: Zpráva nekontrolována \
                (velikost větší než MESSAGE_SIZE_SPAM_MAX)
# ---------------------------------------------------------------------
# Antivirová kontrola
# Vyžaduje nastavení parametrů 'av_scanner' v hlavní sekci.
# #deny
# message = Tato zpráva obsahuje virus ($malware_name)
# demime = *
# malware = */defer_ok
# ---------------------------------------------------------------------

# Volání nástroje SpamAssassin pro získání hodnot $spam_score
# a $spam_report.
# V závislosti na klasifikaci obsahuje proměnná $acl_m9 hodnotu
# „ham" nebo „spam".
# Pokud je zpráva klasifikována jako spam a proměnná $acl_m0 není
# nastavena pro přijetí zprávy, je odmítnuta.
# warn
set acl_m9 = ham

# Pokud se využívají individuální nastavení softwaru SpamAssassin pro
# jednotlivé uživatele, odkomentujte následující řádek a zakomentujte
# „spam = mail".
# Předává se uživatelské jméno, které je součástí adresy příjemce,
# tj. část před prvním výskytem znaku „=" nebo „@", převedená na malá
# písmena. Zpráva by díky předchozímu omezení již neměla mít více
# příjemců.
# spam = ${lc:${extract{1}{=@}{$recipients}{$value}{mail}}}

# ---------------------------------------------------------------------

spam = mail
set acl_m9 = spam
control = fakereject
logwrite = :reject: Odmítnutý spam (score $spam_score): $spam_report

# Přidání záhlaví X-Spam-Status: do zprávy.
```

```
#
warn
message = X-Spam-Status: \
          ${if eq {$acl_m9}{spam}{Yes}{No}} (score $spam_score)\
          ${if def:spam_report {: $spam_report}}
logwrite = :main: Klasifikováno jako $acl_m9 (score $spam_score)
# --- KONEC KONFIGURACE EXISCAN --
# Akceptace zprávy.
# accept
```

Slovník

Ve slovníku jsou uvedeny definice některých slov a termínů, použitých v rámci tohoto dokumentu.

Bayesův filtr

Filtr, který přiřazuje pravděpodobnost, že zpráva je spam, na základě obsahu slov (lépe řečeno na základě výskytu slov nebo frází) z jednotlivých zpráv. Filtr je nejprve nutné vytrénovat tak, že se mu dají k dispozici zprávy nevyžádané pošty a zprávy legitimní pošty. Filtr pak přidělí každému slovu (frázi) ve zprávách hodnocení s tím, jestli se dané slovo nebo fráze vyskytuje v legitimní nebo nevyžádané poště. Slovo spolu se svým hodnocením je uloženo v *Bayesově indexu*.

Tyto filtry mohou zachytit takové příznaky nevyžádané pošty, které jsou programátory opomíjeny při vytváření filtrů založených na seznamech klíčových slov. Bayesovy filtry jsou vždy specificky nastaveny pro jazyk, ve kterém se provádí trénink, a navíc se nastavení může pro jednotlivé uži-vatele lišit. Proto jsou vhodné pro individuální filtrování zpráv jednotlivých uživatelů víc než pro celoplošné filtrování zpráv v rámci celé domény.

Spameři však našli způsob, jak tyto filtry obejít tím, že do svých zpráv přidávají náhodná slova ze slovníku nebo celé části textu. To snižuje pravděpodobnost, že filtr ohodnotí zprávu jako spam, a v dlouhodobějším měřítku degraduje kvalitu Bayesova filtru.

Více informací naleznete na adrese *http://www.everything2.com/index.pl?node=Bayesian.*

Collateral damage – zavlečené škody

Blokování legitimních odesílatelů díky záznamům v seznamech zakázaných položek systému DNS. Některé seznamy (například seznam SPEWS) obvykle zařadí do seznamu celý adresní prostor poskytovatele připojení v případě, kdy mají správci seznamu pocit, že poskytovatel nereaguje odpovídajícím způsobem na stížnosti, což má vliv na *všechny* jeho zákazníky.

Collateral spam – zavlečený spam

Automaticky generované zprávy, které jsou zasílány jako odezva na původní zprávu (obvykle nevyžádanou poštu nebo malware) v případě, kdy je adresa odesílatele falešná. Typickými pří-klady zavlečeného spamu jsou falešná oznámení o viru nebo jiná oznámení o stavu doručení.

Delivery status notification – oznámení o stavu doručení

Zpráva automaticky generovaná agentem přenosu zpráv nebo agentem doručení zpráv, která informuje odesílatele původní zprávy o stavu doručení. Oznámení o stavu doručení může napří-klad informovat odesílatele původní zprávy o tom, že původní zprávu nebylo možné doručit díky nějakým dočasným nebo trvalým problémům, anebo o tom, jak dlouho ještě budou případně další pokusy o doručení pokračovat.

Domain Name System

Standard pro získávání informací o názvech domén v prostředí Internetu. Příklady takových infor-mací jsou například adresy IP serverů (tzv. záznamy typu A), adresy poštovních serverů (záznamy typu MX), obecné informace o serverech (záznamy typu SRV) nebo jiné textové informace (zá-znamy typu TXT). Systém DNS je distribuovaný a hierarchický. Každý název domény má přiřazen jeden nebo více serverů služby DNS, které poskytují informace o této doméně – včetně delego-vání názvové služby pro poddomény. Doménu nejvyšší úrovně „org“ spravuje registr The Public Internet Registry. Servery služby DNS delegují dotazy pro doménu „tldp.org“ na specifické názvo-vé servery domény The Linux Documentation Project. Názvové servery této domény pak mohou nebo nemusí delegovat dotazy na domény třetí úrovně (např. *www.tldp.org*).

Vyhledávání v systému DNS obvykle provádějí směrující názvové servery, například ty, které jsoupřiděleny poskytovatelem připojení k Internetu.Oznámení o stavu doručení jsou zasílána s prázdnou adresou odesílatele.

Envelope recipient – adresa příjemce

Adresa příjemce, na kterou je zpráva doručena. Adresa je uvedena ve zprávě v průběhu transak-ce protokolu SMPT v příkazu RCPT TO:. Tato adresa může být odlišná od adres uvedených v záhlaví „To:“ a „Cc:“ zprávy.

Envelope sender – adresa odesílatele

Adresa odesílatele, která je uvedena ve zprávě v průběhu transakce protokolu SMTP v příkazu MAIL FROM:. Tato adresa může být odlišná od adresy uvedené v záhlaví „From:“ vlastní zprávy. Speciálním případem je oznámení o stavu doručení, u něhož je

adresa odesílatele prázdná. Tím se zabrání vzniku zacyklených zpráv a umožní se odlišení tohoto typu zpráv od ostatních zpráv.

False negative – falešná negativní detekce

Nevyžádaná pošta (spam, virus nebo malware), která je chybně klasifikovaná jako legitimní pošta, a tudíž nezablokovaná a nefiltrovaná.

False positive – falešná pozitivní detekce

Legitimní pošta, která je chybně klasifikovaná jako spam, a tudíž zablokovaná a filtrovaná.

Fully Qualified Domain Name – úplný název domény

Úplný, globálně jedinečný internetový název domény, například *www.yahoo.com.* Úplný název domény většinou neodkazuje na jediný hostitelský systém. Služby typu www obvykle odkazují na mnoho adres IP, aby bylo možné na jednotlivých serverech provádět vyvažování zátěže. Primár-ní název hostitelského systému by měl být vždy jedinečný, například p16.*www.scd.yahoo.com.* Úplný název domény vždy obsahuje znak tečka („.“). První část názvu před první tečkou je tzv. neúplný název a není globálně jedinečný.

Joe Job

Spam, který vypadá tak, že pochází z platné adresy, obvykle pokus o získání informací nebo pokus o poškození vlastníka takto podvržené adresy. Viz také *http://www.everything2.com/ index.pl?node=Joe%20Job.*

Mail Delivery Agent – agent doručení zpráv

Software, který je spuštěn na počítači, kde je umístěna poštovní schránka uživatele, a který zajišťuje doručení zpráv do schránky. Doručení zprávy se provádí agentem přenosu zpráv, který slouží následně jako agent doručení zpráv. Mezi agenty doručení zpráv patří například Delivery, Procmail, Cyrmaster nebo Cyrdeliver (ze sady nástrojů Cyrus IMAP).

Mail Exchanger – poštovní server

Server vyhrazený pro příjem a odesílání poštovních zpráv pro danou internetovou doménu. Zóna systému DNS pro danou internetovou doménu obvykle obsahuje seznam úplných názvů systémů, které pracují jako poštovní servery pro danou doménu. Každý takový poštovní server je identifi-kován záznamem typu MX a také svou prioritou v případě, kdy poštu pro doménu obsluhuje více poštovních serverů. Server s nejnižší prioritou je považován za primární poštovní server pro danou doménu.

Mail Loop – zacyklení zpráv

Situace, kdy jedna automaticky generovaná zpráva způsobí vygenerování jiné zprávy, která spus-tí vygenerování další zprávy atd. Příkladem může být například mailinglist, v němž jeden z uživa-telů má adresu vlastního mailinglistu. V takových případech se většinou využívá záhlaví „X-Loop:“ a zprávy, které takové záhlaví obsahují, již nejsou zpracovávány.

Mail Transport Agent – agent přenosu zpráv

Software, který je spuštěn na poštovním serveru dané internetové domény a zasílá a přijímá zprá-vy z jiných hostitelských systémů. Mezi agenty přenosu zpráv patří například Sendmail, Postfix, Exim nebo Smail.

Mail User Agent – klientský software

Software, který používá uživatel pro přístup, stahování, čtení a odesílání zpráv. Mezi klientský soft-ware patří například Microsoft Outlook nebo Outlook Express, Apple Mail.app, Mozilla Thunder-bird nebo Ximian Evolution.

Mikropayment schemes – mikroplatební systémy

Odesílatel zprávy využije prostředky poštovního serveru pro vytvoření virtuální poštovní známky pro každého příjemce zprávy. Obvykle se to realizuje vyřešením nějaké matematické úlohy, která vyžaduje velký počet paměťových operací, ale je relativně nenáročná na výkon procesoru. Známka je přidána do záhlaví zprávy a příjemce může provést ověření známky jednodušší dekódovací operací. Základní myšlenka tohoto systému je, že díky nutnosti použít poštovní známku pro kaž-dou adresu příjemce by hromadné odesílání spamu tisícům uživatelů bylo poněkud výpočetně „náročné“.

Mezi mikroplatební systémy patří například:

Camram (viz *http://www.camram.org/*)
Penny Black Project firmy Microsoft. (*http://research.microsoft.com/research/sv/ PennyBlack/*)

Open Proxy – otevřený proxy systém

Proxy systém, který akceptuje připojení protokolu TCP/IP odkudkoliv a přesměruje je kamkoliv. Takové systémy jsou obvykle zneužívány spamery, kteří je využívají pro zakrytí vlastní adresy IP a efektivnější distribuci nevyžádaných zpráv mezi mnoho hostitelských systémů a sítí.

Open Relay – systém, který umožňuje přeposílání zpráv

Systém, který akceptuje poštovní zprávy od kohokoliv a přesměrovává je kamkoliv. V roce 1980 byl takřka každý poštovní server

typu open relay. Zprávy obvykle předtím, než byly doručeny pří-jemcům, procházely přes více hostitelských systémů. V současnosti jsou legitimní zprávy od ode-sílatelů téměř výhradně zasílány odchozími agenty přenosu zpráv a doručovány poštovním serve-rům domén příjemců. Servery Open Relay existují i v současnosti a jsou téměř výhradně zneuží-vány spamery pro zakrytí vlastní identity a pro vyvažování zátěže při rozesílání miliónů zpráv v co největším objemu předtím, než seznamy zakázaných položek systému DNS stihnou zaregistrovat všechny takové Open Relay servery.

Proxy

Počítač, který provádí úkoly v zastoupení někoho jiného. Může například směrovat požadavky protokolu HTTP nebo připojení protokolu TCP/IP do nebo z Internetu. Velké organizace nebo někdy i celé země používají webové proxy pro filtrování odchozích požadavků protokolu HTTP z vnitřních sítí. Tento proces může nebo nemusí být pro koncové uživatele transparentní.

Ratware

Software používaný spamery pro hromadné rozesílání nevyžádané pošty, který je navržen s cílem doručit co možná největší objem zpráv ve velice krátkém čase. Většina softwaru tohoto typu obsa-huje pouze nejnutnější implementaci protokolu SMTP, která umí doručit zprávy pouze v ideálním případě. Implementace zahrnuje poskytování falešných nebo nepřesných informací v transakcích protokolu SMTP. Ratware nečeká na odezvy z poštovního serveru a odpojí se v případě, kdy neobdrží odezvu po pár sekundách. Ratware nedodržuje normální mechanismus doručování v pří-padě dočasného selhání služby.

Relay – systém pro přeposílání zpráv

Počítač, který provádí přeposílání zpráv obvykle do nebo z Internetu. Příkladem může být například vyhrazený systém poskytovatele, přes který zasílají odchozí zprávy všichni zákazníci poskytovatele.

Request for Comments – dokumenty RFC

Definice, která pochází z adresy *http://www.rfc-editor.org/*, říká: „Sada dokumentů RFC je sada technických a organizačních poznámek na téma Internet. […] Poznámky v dokumentech RFC obsahují různé aspekty počítačových sítí, včetně protokolů, procedur, programů a metod a jiných poznámek, názorů a také humoru."
Dokumenty obsahují kromě jiného i popisy protokolů a formátů dat. O doručování zpráv pojednávají následující dokumenty:

RFC 2821 „Simple Mail Transfer Protocol"
RFC 2822 „Internet Message Format"

Spam Trap – spamová past

Adresa zpráv elektronické pošty, která je nastražena tak, aby ji mohli snadno získat roboti, kteří sbírají adresy v prostředí Internetu. Obvykle se používají jako zdroj spamu pro seznamy zakáza-ných položek systému DNS nebo pro úložiště signatur zpráv nevyžádané pošty. Zprávy zaslané na tuto adresu jsou v naprosté většině pouze spam nebo malware. Některé takové zprávy však mohou být zavlečený spam, který obsahuje například oznámení o stavu doručení s podvrženými adresami odesílatele. Pokud spamová past neobsahuje mechanismus, jak odfiltrovat takové zprá-vy, může být jako zdroj spamu nespolehlivá.

Zombie host – hostitelský systém – zombie

Hostitelský systém s připojením k Internetu infikovaný virem nebo červem, který rozesílá zprávy elektronické pošty. Takové hostitelské systémy obvykle využívají operační systém Microsoft Win-dows a obvykle jsou to „domácí" uživatelé. Vlastníci těchto systémů obvykle nemají tušení o tom, že je systém infikovaný, a jejich poskytovatelé připojení k Internetu obvykle nepodniknou nic pro jejich odpojení.

Naštěstí existují seznamy zakázaných položek systému DNS, jako je např. „dul.dnsbl.sorbs.net", který obsahuje bloky adres takových „domácích" uživatelů. Tento seznam je vhodné využít pro odmítání příchozích zpráv. Legitimní zprávy od takových uživatelů obvykle zasílají vyhrazené poštovní servery poskytovatelů.

Ipv6 v Linuxu

Obecně

Účelem tohoto návodu je odpovědět na základní i zasvěcené otázky o IPv6 v operačním systému Linux. Tento návod poskytne čtenáři dostatek informací k tomu, aby mohl systém nainstalovat, zkonfigurovat a používat aplikace IPv6 na linuxových strojích.

Informace o překladech naleznete na adrese *http://tldp.org/HOWTO/Linux+IPv6-HOWTO/general-translations.html.* Seznam změn najdete na *http://tldp.org/HOWTO/Linux+IPv6-HOWTO/revision-history.html.*

Napsal a autorská práva vlastní Peter Bieringer. Autora je možno kontaktovat elektronickou poštou na adrese <*pb@bieringer.de*> a také prostřednictvím jeho stránky *http://www.bieringer.de/pb/.*

Předmluva

Pár slov úvodem:

Kolik verzí návodu k Linuxu & IPv6 koluje po Internetu?

Včetně tohoto 3. Jestli se vám to zdá mnoho, omlouvám se.

Linux IPv6 FAQ/návod (nepoužívaný)

První dokument na téma IPv6 napsal *Eric Osborne* a jmenoval se Linux IPv6 FAQ/HOWTO (*http://www.linuxhq.com/IPv6/*, prosím, používejte jej pouze z historických důvodů). Poslední verze 3.2.1 byla vydaná 14. července 1997.

Prosím o pomoc: Zná-li někdo datum narození tohoto návodu, pošlete mi, prosím, e-mail (tuto informaci potřebuji do „historie“).

IPv6 & Linux – návod (udržovaný)

Druhá verze se jmenuje IPv6 & Linux – HowTo (*http://www.bieringer.de/linux/IPv6/*), napsal jsem ji já (*Peter Bieringer*) v čistém HTML. Vznikla v dubnu 1997 a první anglická verze vyšla v červnu 1997. Budu pokračovat v její údržbě, avšak ve srovnání s Návodem k systému Linux IPv6, který právě čtete, bude pomalu zastarávat (ne však úplně).

Návod k systému Linux IPv6 (tento dokument)

Vzhledem k tomu, že IPv6 & Linux – HowTo (*http://www.bieringer.de/linux/IPv6/*) je napsán v čistém HTML, není kompatibilní s The Linux Documentation Project (TLDP, *http://www.tldp.org/*). Koncem listopadu 2001 jsem (*Peter Bieringer*) byl požádán o přepracování IPv6 & Linux – HowTo (*http://www.bieringer.de/linux/IPv6/*) do SGML. Avšak vzhledem k nedostatečné návaznosti tohoto návodu (Future of IPv6 & Linux – HowTo, *http://www.bieringer.de/linux/IPv6/IPv6-HOWTO/IPv6-HOWTO-0.html#history*) a vzhledem k tomu, že IPv6 se stává standardem, rozhodl jsem se napsat nový dokument, který by obsahoval veškeré základní i detailní informace, jež budou platné i do budoucna. Současně jsem do jisté míry aktualizoval i obsah druhého návodu v pořadí IPv6 & Linux – HowTo (*http://www.bieringer.de/linux/IPv6/*).

Pojmy, vysvětlivky a zkratky Sí

Base 10 (základ 10)

: Desítková soustava, číselné hodnoty se vyjadřují číslicemi 0–9.

Base 16 (základ 16)

: Používá se v nižších i vyšších programovacích jazycích, také se nazývá hexadecimální (šest-náctková) soustava, číselné hodnoty se vyjadřují číslicemi 0–9 a znaky A–F (je jedno, zda malý-mi nebo velkými).

Base 85 (základ 85)

: Číselné hodnoty se vyjadřují 85 různými číslicemi a znaky, takže řetězce jsou kratší. V praxi jsem však toto vyjádření ještě nikdy neviděl.

Bit

: Nejmenší paměťová jednotka, nabývá hodnot 1 (zapnuto/pravda) nebo 0 (vypnuto/nepravda).

Byte (bajt)

: Většinou 8 bitů (nikoli však nutně – viz starší počítačové systémy).

Device (zařízení)

Zde je to hardware síťového připojení, viz také NIC.

Dual homed host (počítač připojený ke dvěma sítím)

Počítač připojený ke dvěma sítím tvoří uzel se dvěma síťovými rozhraními (na fyzickou nebo virtuální síť) určenými pro dvě různá připojení, který však mezi těmito rozhraními nepřenáší žádné pakety.

Host (počítač)

Obecně počítač s jedním připojením k síti. Obvykle má jedno aktivní síťové rozhraní, tj. Ethernet, nebo (nikoli „a“) PPP (point-to-point, dvoubodové spojení).

Interface (rozhraní)

Většinou totéž, co „zařízení“, viz také NIC.

IP Header (IP hlavička)

Hlavička IP paketu (každý paket v síti má hlavičku závislou na síťové vrstvě).

Link (připojení)

Připojení je přenosové médium pro přenos síťových paketů druhé vrstvy, například Ethernet, Token Ring, PPP, SLIP, ATM, ISDN, Frame Relay...

Node (uzel)

Uzel je tvořen počítačem nebo směrovačem.

Octet (oktet, osmice)

8 bitů, zhruba totéž co „bajt“.

Port

Informace pro dispečera TCP/UDP (4. vrstva) o přenosu do vyšších vrstev.

Protocol (protokol)

Každá síťová vrstva většinou obsahuje pole protokolů, které jí pomáhá s dispečinkem přená-šených dat z vrstvy 2 (MAC, spojové) a 3 (IP, síťové) do vyšších vrstev.

Router (směrovač)

Směrovač je uzel se dvěma nebo více síťovými rozhraními (fyzických nebo virtuálních sítí), který předává pakety mezi těmito rozhraními.

Soket

IP soket je definován zdrojovými a cílovými IP adresami a porty (a spojením).

Stack (zásobník)

Soubor síťových vrstev.

Subnetmask (podmaska sítě)

IP sítě používají bitové masky k oddělení lokálních a vzdálených sítí.

Tunnel (tunel)

Tunel je obvykle dvoubodové spojení pro přenos paketů s daty v různých protokolech, napří-klad tunel z IPv6 do IPv4.

Zkratky

ACL

Access Control List (seznam pro řízení přístupů).

API

Application Programming Interface (aplikační programové rozhraní).

ASIC

Application Specified Integrated Circuit (integrovaný obvod pro aplikaci).

BSD

Berkeley Software Distribution (distribuce Unixu).

CAN-Bus

Controller Area Network Bus (systém fyzických sběrnic).

ISP

Internet Service Provider (poskytovatel připojení k Internetu).

KAME

Projekt – společné úsilí šesti japonských společností o vytvoření volného zásobníku adres IPv6 a IPsec (pro IPv4 i pro IPv6) pro varianty systému BSD, *http://www.kame.net/*.

LIR

Local Internet Registry (lokální internetový registr).

NIC

Network Interface Card (síťová karta).

RFC

Request For Comments – množina technických a organizačních poznámek o Internetu.

USAGI

UniverSAl playGround for Ipv6 Project – práce, jejichž účelem je zajistit kvalitní zásobník protokolů IPv6 pro systémy Linux.

Další informace Pokračovací znak dlouhého řádku

Kvůli lepší viditelnosti v souborech PDF a PS se pro signalizaci zalomených řádků používá speci-ální znak „ “.

Zástupné symboly

V obecných příkladech někdy naleznete něco jako:

```
<myipaddress>
```

Aby byly příkazové řádky nebo skripty ve vašem systému funkční, je nutno tyto zástupné symboly nahradit náležitým obsahem (pochopitelně bez znaků < a >). Výsledkem bude například:

```
1.2.3.4
```

Příkazy v shellu

Příkazy proveditelné běžným uživatelem (nikoli root) začínají znakem $, např.:

```
$ whoami
```

Příkazy proveditelné uživatelem root začínají znakem #, např.:

```
# whoami
```

Předpoklady k užívání návodu Personální předpoklady

Zkušenosti s unixovými nástroji

Měli byste být schopni používat hlavní unixové nástroje jako grep, awk, find, ... a znát jejich běžně používané volby na příkazovém řádku.

Zkušenosti s teorií sítí

Měli byste něco vědět o vrstvách, protokolech, adresách, kabelech, zásuvných modulech atd. Pokud v této oblasti teprve začínáte, můžete si doplnit znalosti například z *http://www.linux-ports.com/howto/intro_to_networking/*.

Zkušenosti s konfigurací IPv4

S konfigurací IPv4 byste určitě měli mít nějaké zkušenosti, jinak bude pro vás dost obtížné pocho-pit, o co jde.

Zkušenosti s DNS

Také byste měli rozumět tomu, co je DNS (Domain Name System).

Zkušenosti se strategiemi ladění v síti

Přinejmenším byste měli umět používat tcpdump a vědět, k čemu je. Jinak pro vás bude ladění na síti složité.

Hardware kompatibilní s operačním systémem Linux

Určitě chcete experimentovat se skutečným hardwarem a ne jen usínat u čtení návodu.

Základy

Co je IPv6?

IPv6 je nový protokol pro vrstvu 3 (viz *http://www.linuxports.com/howto/intro_to_networking/ c4412.htm#PAGE103HTML*), který nahrazuje IPv4 (známý také pod jménem IP). IPv4 byl navržen už dávno (RFC 760/Internet Protocol z ledna 1980) a už od jeho počátků se množily požadavky na zvět-šení počtu adres a rozšíření funkcionality. Nejnovějším protokolem RFC je RFC 2460/Internet Proto-col Version 6 Specification. Nejdůležitější změnou je v něm nový tvar hlavičky pro adresový prostor zvětšený z 32 bitů na 128 bitů. Vzhledem k tomu, že vrstva 3 je zodpovědná za přenos paketů na základě adres, musí obsahovat nové adresy IPv6 (zdrojovou i cílovou) podobně jako původně adre-sy IPv4.

Další informace o historii IPv6 naleznete v odpovídajícím RFC například na adrese *http://www.switch.ch/lan/ipv6/references.html.*

Historie IPv6 v Linuxu

Historie IPv6 v letech 1992, 1993 a 1994 je v hlavních rysech popsána v *http://www.laynet works.com/IPv6.htm#CH3.*

Zbývá doplnit: podrobnější časové a věcné údaje ...

Počátek

První kód, který se vztahoval k IPv6, doplnil do linuxového jádra 2.1.8 Pedro Roque v listopadu 1996. Byl založen na BSD API:

diff -u --recursive --new-file v2.1.7/linux/include/linux/in6.h
linux/include/linux/in6.h --- v2.1.7/linux/include/linux/in6.h Thu Jan 1 02:00:00 1970 +++ linux/include/linux/in6.h Sun Nov 3 11:04:42 1996 @@ -0,0 +1,99 @@ +/* + * Types and definitions for AF_INET6 + * Linux INET6 implementation + * + * Authors: + * Pedro Roque <******> + * + * Source: + * IPv6 Program Interfaces for BSD Systems + * <draft-ietf-ipngwg-bsd-api-05.txt>

Tyto řádky byly okopírovány ze záplaty 2.1.8 (e-mailová adresa byla přepsána).
Mezitím

Vzhledem k nedostatku pracovních sil nedržela implementace jádra krok s návrhy nových vydá-ní RFC. V říjnu 2000 byl v Japonsku zahájen projekt USAGI (*http://www.linux-ipv6.org/*), jehož cílem bylo implementovat veškerou chybějící nebo zastaralou podporu IPv6 v Linuxu. Projekt sle-doval průběžnou implementaci IPv6 ve FreeBSD v rámci projektu *http://www.kame.net/.* Občas provedli porovnání se zdrojovým kódem jádra vanilla Linuxu.

Současnost

Záplata USAGI je bohužel tak velká, že není možné ji zahrnout do zdrojového kódu linuxového jádra řady 2.4.x. Tato řada tak postrádá některá (nebo dokonce mnohá) rozšíření a ani neodpoví-dá současným návrhům RFC (viz *http://www.ietf.org/html.charters/ipv6-charter.html*). To může způsobit určité problémy při součinnosti s jinými operačními systémy.

Budoucnost

Projekt USAGI využívá probíhajícího vývoje jádra řady 2.5.x k tomu, aby do něj přidal veškerá svá rozšíření. Jádro řady 2.6.x už obsahuje úplnou a aktuální implementaci IPv6.

Jak vypadají adresy IPv6

Jak už jsme se zmínili, adresa IPv6 má 128 bitů. To je počet, který odpovídá velmi vysokým desítkovým číslům s 39 číslicemi:

2^128-1: 340282366920938463463374607431768211455

Taková čísla nejsou adresy, které bychom si byli schopni zapamatovat. Adresové schéma IPv6 je bitově orientováno (podobně jako IPv4, což ale není zcela zřejmé). Proto je lépe zapisovat tako-vá velká čísla hexadecimálně. V hexadecimálním tvaru odpovídají 4 bity (polovina bajtu, říká se jim také „nibble“) číslici nebo znaku v rozsahu 0–9 a a–f (10–15). Tento tvar redukuje délku adre-sy IPv6 na 32 znaků.

2^128-1: 0xffffffffffffffffffffffffffffffff

Tento tvar ještě úplně nevyhovuje (možná záměna nebo přehlédnutí hexadecimální číslice), takže návrháři IPv6 vymysleli hexadecimální tvar s dvojtečkou jako oddělovačem šestnáctic bitů. Navíc odstranili úvodní „0x“ (označení hexadecimálních hodnot používané v programovacích jazycích):

2^128-1: ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff

Adresa pak může vypadat například takto (viz typy adres dále):

3ffe:ffff:0100:f101:0210:a4ff:fee3:9566

Úvodní nuly v každé šestnáctici lze pro zjednodušení vynechat:

3ffe:ffff:0100:f101:0210:a4ff:fee3:9566 -> ¬ 3ffe:ffff:100:f101:210:a4ff:fee3:9566

Jednu posloupnost šestnáctic, které obsahují pouze nuly, lze zapsat jako „::“. Avšak nejvýše jednu, při větším počtu by zápis už

nebyl jednoznačný.

3ffe:ffff:100:f101:0:0:0:1 -> 3ffe:ffff:100:f101::1

Největší redukce lokální adresy IPv6:

0000:0000:0000:0000:0000:0000:0000:0001 -> ::1
Existuje také tzv. *compact* (soustava se základem 85) definovaná v RFC 1924 (vydaná 1. dubna 1996). V praxi jsem ji ještě neviděl, (nejspíš jde o aprílový žert), ale zde je příklad:

ipv6calc --addr_to_base85 3ffe:ffff:0100:f101:0210:a4ff:fee3:9566 Itu&-ZQ82s>J%s99FJXT

Informace: Kalkulátor a konverzní program ipv6calc na adresový formát IPv6 naleznete zde:
http://www.deepspace6.net/projects/ipv6calc.html.

Často kladené otázky (základní) Proč je následníkem IPv4

IPv6a nikoli IPv5?

V hlavičce IP adresy jsou první čtyři bity vyhrazeny verzi protokolu, takže teoreticky může být číslo protokolu 1 až 15:

4: se používá pro IPv4,
5: je vyhrazeno pro Stream Protocol (STP, RFC 1819), který nebyl nikdy používán.

První volné číslo bylo 6 – a IPv6 byl na světě!

Adresy IPv6: proč tolik bitů?

Při návrhu IPv4 se mnozí domnívali, že 32 bitů postačí pro celý svět. Stačily dodnes a možná bystačily ještě pár let. 32 bitů však nestačí k tomu, aby v budoucnosti byla každému síťovému zaří-zení přidělena síťová adresa. Když vezmete mobilní telefony, auta (včetně elektronických zařízenína sběrnici CAN-bus), topinkovače, ledničky, spínače osvětlení atd.
Návrháři tedy zvolili 128 bitů, což je číslo čtyřikrát delší a 2^96krát větší než dnešní IPv4.Využitelnost je ovšem menší, než se může zdát. To proto, že v současném adresovém schématuje 64 bitů využito pro identifikátor rozhraní a zbývajících 64 bitů pro směrování. Vezmeme-li
v úvahu současnou omezenou úroveň agregace (/48, /32, ...), je docela možné, že ani 128 bitůnebude stačit. Doufejme však, že už se toho nedožijeme.Další informace viz RFC 1715 a RFC 3194 .

Adresy IPv6: proč tak málo bitů?

I když na Internetu jsou (možná) lidé (vím jen o Jimovi Flemingovi...), kteří už uvažují o IPv8 a IPv16, jejich návrhy mají ještě daleko ke schválení a k implementaci. 128 bitů je prozatím nej-lepší volbou z hlediska režie hlavičky a přenosu dat. Když vezmete minimální MTU (Maximum Transfer Unit) v IPv4 (576 oktetů) a v IPv6 (1 280 oktetů), délka hlavičky v IPv4 je 20 oktetů (mini-málně, s volbami může vzrůst až na 60) a v IPv6 je 48 oktetů (pevná). To je 3,4 % MTU v IPv4 a 3,8 % v IPv6. Znamená to, že režie je prakticky stejná. Víc bitů pro adresu by vyžadovalo větší hlavičky, a tedy větší režii. Uvážíme-li tedy maximální MTU u běžných připojení (např. Ethernet): 1 500 oktetů (ve speciálních případech: 9k oktetů při použití Jumbo Frame). Používat 10 až 20 % přenášených dat v paketech vrstvy 3 pro adresy a jen zbytek pro užitečný náklad by nebylo dobré.

Typy adres

Podobně jako v IPv4 lze adresy IPv6 rozdělit pomocí masky na síťovou část a počítačovou část. V IPv4 se ukázalo, že někdy by bylo příjemné, kdyby jednomu rozhraní bylo možné přidělit více než jednu IP adresu, každou pro jiný účel (aliasy, multicast). S ohledem na použitelnost i v budoucnosti šla IPv6 dál a umožňuje připojit ke každému rozhraní více než jednu adresu IPv6.
V RFC neexistuje žádné omezení, pouze v implementaci zásobníku IPv6 (prevence útoků typu

DoS – odmítnutí služby).Vzhledem k velkému počtu bitů určených pro adresu jsou v IPv6 využity některé úvodní bity natypy adres a doufejme, že nebudou dále rozděleny jako v případě tříd A, B a C v současné adre-se IPv4.

Bity jsou tedy rozděleny na síťovou část (horních 64 bitů) a počítačovou část (spodních 64 bitů)
kvůli snazší autokonfiguraci.

Adresy bez speciálního prefixu Adresy lokálních počítačů

Toto je speciální loopback adresa podobná adrese „127.0.0.1“ v IPv4. V IPv6 má tvar:

0000:0000:0000:0000:0000:0000:0000:0001

anebo ve zkráceném tvaru:

::1

Pakety s touto zdrojovou nebo cílovou adresou by nikdy neměly opustit odesilatele.

Nespecifikovaná adresa

Toto je speciální adresa „kohokoli“, resp. „0.0.0.0“ v IPv4. V IPv6 je to:

0000:0000:0000:0000:0000:0000:0000:0000

resp.:

::

Tyto adresy se většinou používají v připojení soketů (k libovolné adrese IPv6) nebo ve směrova
cích tabulkách.
Poznámka: Nespecifikovanou adresu nelze použít jako cílovou adresu.

Adresa IPv6 s vestavěnou adresou IPv4

Existují dva druhy adres, které obsahují adresu IPv4.

IPv4-mapped IPv6 address

IP adresa kompatibilní s adresou IPv6 se někdy používá pro sokety vytvořené démonem aktuali-zovaným pro IPv6, avšak s vazbou pouze na adresu IPv4.Tyto adresy jsou definované speciálním prefixem o délce 96 (a.b.c.d je adresa IPv4):

0:0:0:0:0:ffff:a.b.c.d/96

anebo ve zkráceném tvaru:

::ffff:a.b.c.d/96

Například adresa IPv4 1.2.3.4 vypadá takto:

::ffff:1.2.3.4

Adresa IPv6 kompatibilní s IPv4

Používá se pro automatické tunelování (RFC 2893), které je nahrazeno tunelováním 6to4 (kapitola „Vytváření tunelů 6to4“).
0:0:0:0:0:0:a.b.c.d/96

nebo ve zkráceném tvaru:

::a.b.c.d/96

Síová část adresy (prefix)

Návrháři definovali pouze některé typy adres a ponechali značný prostor pro budoucí definice plynoucí z nově vznikajících požadavků. Adresové schéma je definováno v RFC 2373, avšak exi-stuje už i nový návrh: ftp://ftp.ietf.org/internet-drafts/.

Podívejme se nyní na různé typy prefixů (a tedy typů adres):

Lokální adresový typ pro připojení

Speciální adresy, které budou platné pouze při napojení na rozhraní. Pokud by taková adresa byla cílovou adresou, paket by nikdy neprošel směrovačem. Používá se pro spojovací komunikaci, například:

Je na tomto spojení někdo jiný?
Má někdo speciální adresu (tj. hledá směrovač)?

Začínají (kde *„x“* je libovolný hexadecimální znak, obvykle *„0“*):

fe8x: <-*v současnosti se používá pouze tento.*fe9x:feax:febx:

Adresa s tímto prefixem se po bezstavové autokonfiguraci nalézá na všech rozhraních s aktivova-ným IPv6 (v jiných případech obvykle nikoli).

Lokální adresový typ pro sí (angl. „site“)

Adresy podobné RFC 1918 v IPv4 s tou výhodou, že je možno využít těchto 16 bitů pro nejvýše

65 536 podsítí. Srovnatelné s 10.0.0.0/8 v IPv4.Další výhoda: Vzhledem k tomu, že rozhraní IPv6 je možné přiřadit více než jednu adresu, jemožné přiřadit této síti kromě globální adresy také lokální adresu.
Začíná od:

fecx: <- nejčastěji používaná.
fedx:
feex:
fefx:

(kde *„x“* je libovolný hexadecimální znak, obvykle *„0“*)
Poznamenejme, že v současnosti probíhají diskuse o zrušení tohoto typu adres, neboť jsou s nímspojeny určité problémy. Další informace viz *http://www.ietf.org/internet-drafts/*.Podle mého skromného názoru laboratorní testy prokázaly, že tyto adresy vyhovují.

Globální adresový typ pro „(agregovatelné) globální jednosměrné vysílání”

V současnosti existuje jediný definovaný globální adresový typ (první návrh nazvaný „provider based“ [založený na poskytovateli] byl před několika lety zamítnut: RFC 1884, některé zbytky po něm naleznete ve zdrojovém kódu starších linuxových jader).

Začíná na (*„x“* jsou hexadecimální znaky):
2*xxx*: 3*xxx*:

Poznámka: Prefix „agregovatelný“ je v současných návrzích opuštěn. Jsou definovány některé další podtypy, viz:

Testovací adresy 6bone

To byly první definované a používané globální adresy. Všechny začínají na:

3ffe:

Příklad:

3ffe:ffff:100:f102::1

Speciální testovací adresa typu 6bone, která nikdy nebude globálně jednoznačná, začíná na:

3ffe:ffff:

a můžete se s ní setkat ve starších příkladech, neboť kdyby v nich byla uvedena skutečná adresa, mohl by si ji někdo přenést do svého konfiguračního souboru, čímž by nechtěně vytvořil dupli-kát globálně jednoznačné adresy. Tím by mohly původnímu počítači vzniknout vážné problémy (např. by mohl dostávat odpovědní pakety na požadavek, který neodeslal). Vzhledem k tomu, že IPv6 se nyní tvoří, tento prefix nebude podporován a po 6. 6. 2006 bude pravděpodobně zcela odstraněn (podrobnosti viz RFC 3701).

Adresy 6to4

Tyto adresy jsou určeny pro speciální tunelové mechanismy RFC 3056 a RFC 2893, kódování adres IPv4 a možné podsítě. Začínají na:

2002:

Například IP adresa 192.168.1.1/5 by vypadala takto:

2002:c0a8:0101:5::1

Z dané adresy IPv4 můžeme vytvořit tuto adresu pomocí krátkého příkazového řádku v shellu:

ipv4=”1.2.3.4”; sla=”5”; printf “2002:%02x%02x:%02x%02x:%04x::1” `echo $ipv4 ¬ | tr “.” “ “` $sla

Viz také kapitoly „Vytváření tunelů 6to4“ a „Online informace“.

Přiděleno poskytovatelem pro hierarchické směrování

Tyto adresy jsou delegovány poskytovatelům připojení k Internetu (ISP) a začínají na:

2001:
Prefixy pro hlavní (páteřní) ISP (někdy se nazývají LIR) jsou delegovány v hlavních regionálníchrejstřících (viz kapitola „On-line informace“) a dostávají prefix dlouhý 32 bitů.Zákazník dostává prefix dlouhý 48 bitů.

Adresy vyhrazené pro příklady a dokumentaci
Pro příklady a dokumentaci jsou běžně vyhrazeny dvě oblasti adres:

3ffe:ffff::/32 2001:0DB8::/32 EXAMPLENET-WF
Tyto oblasti adres by měly být filtrované na základě zdrojových adres a pokud možno by NEMĚLY být směrovány na hraniční směrovače Internetu.

Adresy pro vícesměrový přenos (multicast)

Vícesměrové adresy se používají pro související služby. Začínají vždy na (*xy* je rozsah):

ffxy:

Jsou rozděleny na rozsahy a typy:

Rozsahy vícesměrového přenosu

Rozsah vícesměrového přenosu je parametr, který udává maximální vzdálenost, kam až může putovat vícesměrový paket od odesilatele. V současnosti jsou definovány následující oblasti (rozsahy):

ffx1: lokální vzhledem k uzlu, paket nikdy neopustí uzel,
ffx2: lokální vzhledem k připojení, pakety nikdy nejsou přeposílány směrovači, takže nikdy neopustí dané připojení,
ffx5: lokální vzhledem k síti, paket nikdy neopustí síť,
ffx8: lokální vzhledem k organizaci, paket nikdy neopustí organizaci (implementace není jednoduchá, je nutno použít směrovací protokol),
ffxe: globální rozsah,
další jsou rezerva.

Vícesměrové typy

Existuje mnoho dalších definovaných a rezervovaných typů (podrobnosti viz RFC 2373). Například:

Adresa všech uzlů: ID = 1h, adresuje všechny počítače na lokálním uzlu (ff01:0:0:0:0:0:0:1) nebo na připojení (ff02:0:0:0:0:0:0:1).
Adresa všech směrovačů: ID = 2h, adresuje všechny směrovače na lokálním uzlu (ff01:0:0:0:0:0:0:2) nebo na připojení (ff02:0:0:0:0:0:0:2) nebo na lokální síti (ff05:0:0:0:0:0:0:2).

Vyžádaná vícesměrová adresa připojení k lokálnímu uzlu

Speciální vícesměrová adresa používaná jako cílová adresa při „zjišťování souseda“, neboť na rozdíl od IPv4 v IPv6 není protokol ARP.

Adresa může vypadat například takto:

ff02::1:ff00:1234

Prefix říká, že je to vícesměrová adresa připojení k lokálnímu uzlu. Přípona je generovaná z cílové adresy. V tomto příkladu by měl být paket poslán na adresu „fe80::1234”, avšak síťový zásobník nezná fyzickou adresu spojové vrstvy (vrstvy 2). Nahradí horní 104 bity číslem „ff02:0:0:0:0:1:ff00::/104” a spodní 24 bity ponechá beze změny. Tato adresa se nyní použije „na připojení“, aby se nalezl odpovídající uzel, který musí poslat odpověď se svojí fyzickou adresou spojové vrstvy.

Výběrové (anycast) adresy

Výběrové adresy jsou speciální adresy používané například k vyhledávání nejbližšího serveru DNS, serveru DHCP nebo podobných dynamických skupin. Adresy jsou přejaty z jednosměrného adresového prostoru (v současnosti agregovatelné globálně nebo v lokální síti). Mechanismus výběrové adresy (z pohledu klienta) zpracovávají dynamické směrovací protokoly.

Poznámka: Výběrové adresy nemohou být zdrojovými adresami, používají se pouze jako cílové adresy.

Výběrová adresa směrovače podsítě

Jednoduchým příkladem výběrové adresy je výběrová adresa směrovače podsítě. Předpokládejme, že uzel má následující globálně přidělenou adresu IPv6:

3ffe:ffff:100:f101:210:a4ff:fee3:9566/64 <- Adresa uzlu

Výběrovou adresu směrovače podsítě vytvoříme úplným vynecháním přípony (poslední 64 významové bity):

3ffe:ffff:100:f101::/64 <- Výběrová adresa směrovače podsítě

Adresové typy (počítačová část)

Pro autokonfiguraci a pro účely mobility bylo rozhodnuto, že ve většině běžných adresových typů se budou pro počítač používat spodní 64 bity. Každá podsíť tak může mít velké množství adres. Na tuto část je nutno pohlížet rozdílně.

Automaticky vypočtená (též nazývaná bezstavová)

Při autokonfiguraci se počítačová část adresy vypočte tak, že se metodou EUI-64 provede konverze fyzické adresy rozhraní (pokud existuje) na jednoznačnou adresu IPv6. Neexistuje-li k tomuto zařízení fyzická adresa (například u virtuálních zařízení), použije se něco jiného (například adresa IPv4 fyzické adresy fyzického rozhraní).

Uvažujme znovu o prvním příkladu:

3ffe:ffff:100:f101:210:a4ff:fee3:9566

zde,

210:a4ff:fee3:9566

je počítačová část adresy a je vypočtena z fyzické adresy síťové karty:

00:10:A4:E3:95:66

pomocí *http://standards.ieee.org/regauth/oui/tutorials/EUI64.html* pro identifikátory EUI-48.

Problém se zachováním soukromí u automaticky počítaných adres a jeho řešení

Vzhledem k tomu, že „automaticky vypočtená“ počítačová část je globálně jednoznačná (pokud ovšem výrobce síťových karet nepřiřadil stejnou adresu více než jedné kartě), je možné daného klienta zjistit, pokud nepoužívá nějakou formu zastoupení.

To je známý problém a řešení bylo provedeno rozšířením definice v RFC 3041 (už existuje i nový návrh: ftp://ftp.ietf.org/internet-drafts/). Občas se na základě statické hodnoty náhodně vygene-ruje nová přípona. Poznámka: Tento způsob má smysl pouze v případě odchozích klientských spojení, zatímco u známých serverů se nepoužívá.

Ruční nastavení

Pro server je pravděpodobně snazší zapamatovat si jednodušší adresy, což lze provést. Rozhraní můžeme přidat další adresu IPv6, tj.:

3ffe:ffff:100:f101::1

Pro ruční přípony jako „::1”, jak je uvedeno v předchozím příkladu, musí být 7. nejvýznamnější bit nastaven na 0 (bit označující globální/lokální u automaticky generovaného identifikátoru). Také některé jiné (jinak nevybrané) bitové kombinace jsou vyhrazeny pro výběrové adresy.

Délky prefixu pro směrování

Ve fázi časného návrhu bylo plánováno používání plně hierarchického přístupu ke směrování, aby se co nejvíce zredukovaly velikosti směrovacích tabulek. Vzhledem k počtu stávajících směrova-cích položek IPv4 v hlavních směrovačích (> 104 tisíce v květnu 2001) se tak u hardwarových smě-rovačů (řízených ASIC, tj. „Application Specified Integrated Circuit”) měly snížit nároky na paměť pro ukládání směrovacích tabulek a měla se zvýšit rychlost (při menším počtu položek je vyhle-dávání rychlejší).

Viděno dnešním pohledem, hierarchické směrování bude navrhováno pro sítě pouze v rámci jed-noho poskytovatele připojení. V případě spojení zajišťovaných více ISP to není možné a také to není možné kvůli problému zvanému multi-homing (podrobnosti viz *http://www.ietf.org/internet-drafts/* a *http://arneill-py.sacramento.ca.us/ipv6mh/*).

Délky prefixů (známé i pod jménem „síové masky“)

Podobně jako v IPv4 existuje i zde směrování cesty po síti. Vzhledem k tomu, že standardní zápis síťové masky v případě 128 bitů nevypadá pěkně, návrháři využili schématu CIDR z Ipv4 (CIDR, RFC 1519), kterým se specifikuje počet bitů IP adresy pro směrování. Zápisu se také říká „lomítková“ (slash) notace.

Příklad:

3ffe:ffff:100:1:2:3:4:5/48

Po expanzi:

Síť:
Síťová maska:

> 3ffe:ffff:0100:0000:0000:0000:0000:0000
>
> ffff:ffff:ffff:0000:0000:0000:0000:0000

Hledání cesty

Za normálních okolností (bez QoS, tj. požadavku na kvalitu služby) hledání ve směrovacích tabulkách dopadne tak, že cesta s nejvýznamnějším počtem adresových bitů znamená, že cesta s největší délkou prefixu se najde nejdříve.

Když například jsou ve směrovací tabulce následující položky (seznam není úplný):

3ffe:ffff:100::/48 :: U 1 0 0 sit1 2000::/3 ::192.88.99.1 UG 1 0 0 tun6to4

Uvedená cílová adresa paketů IPv6 bude nasměrována přes dané zařízení

3ffe:ffff:100:1:2:3:4:5/48 -> nasměrováno přes zařízení sit1 3ffe:ffff:200:1:2:3:4:5/48 -> nasměrováno přes zařízení tun6to4

Kontrola systému na IPv6

Než začnete používat IPv6 na linuxovém počítači, musíte otestovat, zda je systém na práci s tímto adresovým schématem připraven. Možná pro to budete muset nejdříve něco udělat.

Jádro a IPv6

Současné linuxové distribuce už obsahují jádra s podporou IPv6. Tuto funkci zajišťuje samostatně překládaný modul, je však možné, že při spuštění systému nebyl automaticky zaveden. Nejaktuálnější informace naleznete na stránce *http://www.bieringer.de/linux/IPv6/status/ IPv6+Linux-status-distributions.html.*

> Poznámka
>
> Už byste neměli používat jádro 2.2.x, neboť skončila aktualizace těch částí, které zpraco
>
> vávají IPv6.

Jak zkontrolovat za chodu, zda jádro podporuje IPv6

Když chcete zkontrolovat, zda stávající jádro podporuje IPv6, podívejte se nejdříve do souboro-vého systému /proc. Musí v něm být položka:

```
/proc/net/if_inet6
```

Krátký automatický test vypadá takto:

```
# test -f /proc/net/if_inet6 && echo „Jádro podporuje IPv6”
```

Když test zhavaruje, je velmi pravděpodobné, že není zaveden modul IPv6.

Zkuste zavést modul IPv6

Modul IPv6 můžete zkusit zavést pomocí

```
# modprobe ipv6
```

Budete-li úspěšní, můžete otestovat jeho zavedení pomocí následujícího kouzelného příkazového řádku:

```
# lsmod |grep -w ‘ipv6’ && echo „Modul IPv6 úspěšně zaveden”
```

A test shora by nyní měl proběhnout úspěšně.

> Poznámka
>
> Bez zavedeného modulu IPv6 může jádro za určitých okolností zhavarovat.

Automatické zavedení modulu

V konfiguračním souboru si můžete nastavit automatické zavádění modulu IPv6. Stačí přidat do konfi-guračního souboru zavaděče jádra (obvykle /etc/modules.conf nebo /etc/conf.modules) řádek:

```
alias net-pf-10 ipv6 # automatické zavedení modulu IPv6
```

Naopak je možné automatické zaváděné modulu IPv6 zablokovat pomocí řádku:

```
alias net-pf-10 off # zablokuj automatické zavedení modulu IPv6
```

> Doplňující poznámka
>
> V jádru řady 2.6.x byl mechanismus zavádění jádra změněn. Nový konfigurační soubor musíte místo /etc/modules.conf pojmenovat /etc/modprobe.conf.

Kompilace jádra s podporou IPv6

Pokud oba výsledky shora byly negativní a jádro nemá podporu IPv6, máte na vybranou z těchto možností:

Aktualizovat distribuci na takovou verzi, aby podporovala IPv6 již v základu (doporučeno začátečníkům).
Přeložit nové jádro vanilla (snadné, pokud víte, jaké potřebujete volby).
Znovu přeložit zdrojový kód jádra, které patří k vaší distribuci (což nemusí být zcela jednoduché).
Přeložit jádro s USAGI.

Rozhodnete-li se přeložit jádro, měli byste s tím mít už určitou zkušenost a také si přečíst *http://www.tldp.org/HOWTO/Kernel-HOWTO.html.*

Nejaktuálnější srovnání mezi jádrem vanilla a rozšířeními jádra USAGI je k dispozici na *http://www.bieringer.de/linux/IPv6/status/IPv6+Linux-status-kernel.html.*

Překlad jádra vanilla

Podrobnější pokyny k překladu jádra s podporou IPv6 naleznete na *http://www.bieringer.de/ linux/IPv6/IPv6-HOWTO/IPv6-HOWTO-2.html#kernel.*

Poznámka

Je-li to možné, měli byste používat jádro řady 2.6.x nebo vyšší, neboť podpora IPv6 v řadě 2.4.x je pouze částečná a používání IPv6 v řadě 2.2.x je beznadějně zastaralé.

Překlad jádra s rozšířeními USAGI

Stejně jako v případě jádra vanilla je překlad tohoto jádra možné doporučit jen pokročilým uži-vatelům, kteří jsou zběhlí v překladech jádra. Viz také projekt USAGI (*http://www.linux-ipv6.org/faq.html*) a *http://www.deepspace6.net/docs/best_ipv6_support.html.*

Síová zařízení s podporou IPv6

Ne všechna stávající síťová zařízení mají už (nebo vůbec) schopnost přenášet pakety IPv6. Hlavní potíž je v tom, že kvůli struktuře síťových vrstev implementace jádra není paket IPv6 ve sku-tečnosti rozpoznáván podle čísla hlavičky (6 místo 4). Je rozpoznáván podle čísla protokolu vrstvy 2 transportního protokolu. Proto žádný transportní protokol, který nepoužívá toto číslo protokolu, nemůže přenášet pakety IPv6. Poznámka: Paket je přenášen spojovou vrstvou, avšak na straně pří-jemce nebude přenos fungovat (můžete si to vyzkoušet například programem tcpdump).

Dosud známá „spojení neschopná přenosu IPv6“

Serial Line IP (SLIP, RFC 1055) by se nyní raději mělo nazývat SLIPv4, jména zařízení: slX.
Parallel Line IP (PLIP), totéž jako SLIP, jména zařízení: plipX.
ISDN se zapouzdřením *rawip*, jména zařízení: isdnX.

Dosud známá „nepodporovaná spojení schopná přenosu IPv6“

- ISDN se zapouzdřením *syncppp*, jména zařízení: ipppX (návrh položky ipppd bude zahr-nut do obecnější vrstvy PPP v jádru řady 2.5.x).

Síové konfigurační nástroje IPv6

Jestliže provozujete jádro s podporou IPv6, avšak nemáte nástroje na konfiguraci IPv6, nedostanete se příliš daleko.

Balík síových nástrojů

Balík síťových nástrojů obsahuje některé nástroje, např. ifconfig nebo route, které vám pomo-hou zkonfigurovat IPv6 na rozhraní. Podívejte se na výstup příkazů ifconfig -? nebo route -?, jest-li se u některé položky objeví IPv6 nebo inet6. Pokud ano, tento nástroj podporuje IPv6.

Zjistíme to pomocí kouzelné formulky:

```
# /sbin/ifconfig -? 2>& 1|grep -qw ‘inet6’ && echo “utilita ‘ifconfig’ ¬ podporuje IPv6”
```

Stejný test můžeme provést v případě route:

```
# /sbin/route -? 2>& 1|grep -qw ‘inet6’ && echo “utilita ‘route’ podporuje ¬ IPv6”
```

Balík iproute

Alexej N. Kuzněcov (člověk odpovědný za údržbu síťového kódu Linuxu) vytvořil množinu nástro-jů pro konfiguraci sítí prostřednictvím síťových zařízení. S touto množinou nástrojů dosáhnete vyšší funkcionalitu, než jakou nabízejí síťové nástroje, avšak není příliš dobře zdokumentovaná a vyža-duje odvážného ducha.

```
# /sbin/ip 2>&1 |grep -qw ‘inet6’ && echo “utilita ‘ip’ podporuje IPv6”
```

Pokud se program /sbin/ip nenajde, vřele doporučuji nainstalovat balík iproute.

Můžete jej získat z vaší linuxové distribuce (pokud jej obsahuje).
Můžete si jej stáhnout v tar balíku z ftp://ftp.inr.ac.ru/ip-routing/ (zrcadlo chybí) a zkom-pilovat jej.
Můžete se pokusit najít odpovídající balík RPM na *http://rpmfind.net/linux/rpm2html/ search.php?query=iproute* (někdy je doporučován překlad balíku SRPMS).

Programy pro testování a ladění podpory IPv6

Poté co jste si připravili systém s podporou IPv6, určitě budete chtít využít IPv6 pro komunikaci po síti. Nejdříve byste se měli naučit prohlížet pakety IPv6 nějakým „očichávacím“ programem. To je velmi důležité, neboť při ladění a odstraňování chyb vám to velmi usnadní diagnostiku.

ping IPv6

Tento program obvykle bývá součástí balíku iputils. Je určen pro jednoduchý přenosový test, který provede tak, že odešle pakety ICMPv6 pro vyžádání ozvěny a čeká na pakety ICMPv6 s odpovědí. Použití:

ping6 <hostwithipv6address> # ping6 <ipv6address> # ping6 [-I <device>] <link-local-ipv6address>

Příklad:

ping6 -c 1 ::1 PING ::1(::1) from ::1 : 56 data bytes 64 bytes from ::1: icmp_seq=0 hops=64 time=292 usec --- ::1 ping statistics ---1 packets transmitted, 1 packets received, 0% packet loss round-trip min/avg/max/mdev = 0.292/0.292/0.292/0.000 ms

Doporučení: ping6 vyžaduje přímý přístup k soketům, musí tedy mít přístupová práva uživatele root. Pokud se tedy běžnému uživateli nedaří spustit ping6, příčiny mohou být dvě:
ping6 není v uživatelské cestě (pravděpodobně proto, že obvykle bývá uložen v /usr/sbin) -> doplnění cesty. Tento postup však nedoporučujeme.
ping6 nepracuje správně zřejmě proto, že nemá přístupová práva uživatele root -> chmod u+s /usr/sbin/ping6.

Specifikace rozhraní pro ping IPv6

Použije-li ping IPv6 lokální adresu připojení, jádro neví, přes které (fyzické nebo virtuální) zařízení musí poslat paket – každé zařízení má lokální adresu připojení. Takový pokus bude mít za následek výpis chybového hlášení:

ping6 fe80::212:34ff:fe12:3456 connect: Invalid argument

V tomto případě musíte specifikovat rozhraní dodatečně, viz:

ping6 -I eth0 -c 1 fe80::2e0:18ff:fe90:9205 PING fe80::212:23ff:fe12:3456(fe80::212:23ff:fe12:3456) from ¬ fe80::212:34ff:fe12:3478 eth0: 56 data bytes 64 bytes from fe80::212:23ff:fe12:3456: icmp_seq=0 hops=64 time=445 usec --- fe80::2e0:18ff:fe90:9205 ping statistics ---1 packets transmitted, 1 packets received, 0% packet loss round-trip ¬ min/avg/max/mdev = 0.445/0.445/0.445/0.000 ms

Ping6 na vícesměrové adresy

Zajímavý způsob, jak zjistit počítače s podporou IPv6 ve spojové vrstvě, je spustit ping6 s lokál-ní vícesměrovou adresou pokrývající všechny uzly:

ping6 -I eth0 ff02::1 PING ff02::1(ff02::1) from fe80:::2ab:cdff:feef:0123 eth0: 56 data bytes 64 bytes from ::1: icmp_seq=1 ttl=64 time=0.104 ms 64 bytes from fe80::212:34ff:fe12:3450: icmp_seq=1 ttl=64 time=0.549 ms (DUP!)

Na rozdíl od IPv4, kde mohou být odpovědi na všesměrovou adresu blokovány, v případě IPv6 takto postupovat nelze (s výjimkou lokálního firewallu IPv6).

traceroute6 IPv6

Tento program je obvykle součástí balíku iputils. Je podobný programu traceroute v IPv4, viz příklad:

traceroute6 www.6bone.net
traceroute to 6bone.net (3ffe:b00:c18:1::10) from 3ffe:ffff:0000:f101::2, 30 hops max, 16 byte packets 1 localipv6gateway (3ffe:ffff:0000:f101::1) 1.354 ms 1.566 ms 0.407 ms 2 swi6T1-T0.ipv6.switch.ch (3ffe:2000:0:400::1) 90.431 ms 91.956 ms 92.377 ms 3 3ffe:2000:0:1::132 (3ffe: 2000:0:1::132) 118.945 ms 107.982 ms 114.557 ms 4 3ffe:c00:8023:2b::2 (3ffe:c00:8023:2b::2) 968.468 ms 993.392 ms 973.441 ms 5 3ffe: 2e00:e:c::3 (3ffe:2e00:e:c::3) 507.784 ms 505.549 ms 508.928 ms 6 www.6bone.net (3ffe:b00:c18:1::10) 1265.85 ms * 1304.74 ms

> Poznámka
>
> Na rozdíl od některých novějších verzí IPv4, jež mohou používat pakety ICMPv4 s vyžá-dáním ozvěny a pakety UDP (implicitně), program traceroute6 může posílat pouze pake-ty UDP. Jak možná víte, pakety ICMP s vyžádáním ozvěny prozatím přijímají spíše firewally a ACL na směrovačích.

tracepath6 IPv6

Tento program je obvykle součástí balíku iputils. Je podobný programu traceroute6, sleduje cestu k danému místu určení a na této cestě vyhledává MTU, viz příklad:

tracepath6 www.6bone.net 1?: [LOCALHOST] pmtu 1480
1: 3ffe:401::2c0:33ff:fe02:14 150.705ms
2: 3ffe:b00:c18::5 267.864ms
3: 3ffe:b00:c18::5 asymm 2 266.145ms pmtu 1280
3: 3ffe:3900:5::2 asymm 4 346.632ms

4: 3ffe:28ff:ffff:4::3 asymm 5 365.965ms
5: 3ffe:1cff:0:ee::2 asymm 4 534.704ms
6: 3ffe:3800::1:1 asymm 4 578.126ms !N Resume: pmtu 1280

tcpdump IPv6

tcpdump je v Linuxu hlavním nástrojem na prohlížení paketů. Příklady najdete v dalším textu.Podpora IPv6 je součástí stávajících vydání verze 3.6.Aby se minimalizoval šum, pro filtrování paketů používá tcdump výrazy:

icmp6: filtry pro provoz skupiny ICMPv6,
ip6: filtry pro provoz skupiny IPv6 (včetně ICMPv6),
proto ipv6: filtry pro provoz skupiny tunelů Ipv6-in-IPv4,
not port ssh: potlačení výpisu paketů SSH v programu tcpdump, který běží na vzdáleném sezení SSH.

Také některé volby v příkazových řádcích jsou užitečné k zachycení a výpisu více informací z paketu, a to zejména z paketů ICMPv6: ▪-s 512: zvětšení prohlížené délky (*snap length*) při prohlížení paketů na 512 bajtů, ▪-vv: velmi upovídaný výstup, ▪-n: nepřevádět adresy na jména, používá se, když nepracuje korektně reverzní DNS.

ping IPv6 na skupinu adres 3ffe:ffff:100:f101::1 prostřednictvím lokálního spojení

```
# tcpdump -t -n -i eth0 -s 512 -vv ip6 or proto ipv6 tcpdump: listening on eth0 3ffe:ffff:100:f101:2e0:18ff:fe90:9205 > 3ffe:ffff:100:f101::1:
icmp6: echo
  request (len 64, hlim 64) 3ffe:ffff:100:f101::1 > 3ffe:ffff:100:f101:2e0:18ff:fe90:9205: icmp6: echo
reply (len 64, hlim 64)
```

ping IPv6 na 3ffe:ffff:100::1 směrovaný přes tunel z IPv6 do IPv4

1.2.3.4 a 5.6.7.8 jsou koncové body tunelů (všechny adresy jsou příklady)

```
# tcpdump -t -n -i ppp0 -s 512 -vv ip6 or proto ipv6
tcpdump: listening on ppp0
1.2.3.4 > 5.6.7.8: 2002:ffff:f5f8::1 > 3ffe:ffff:100::1: icmp6: echo request (len 64, hlim 64) (DF) (ttl 64, id 0, len 124)
5.6.7.8 > 1.2.3.4: 3ffe:ffff:100::1 > 2002:ffff:f5f8::1: icmp6: echo reply (len 64, hlim 61) (ttl 23, id 29887, len 124)
1.2.3.4 > 5.6.7.8: 2002:ffff:f5f8::1 > 3ffe:ffff:100::1: icmp6: echo request (len 64, hlim 64) (DF) (ttl 64, id 0, len 124)
5.6.7.8 > 1.2.3.4: 3ffe:ffff:100::1 > 2002:ffff:f5f8::1: icmp6: echo reply (len 64, hlim 61) (ttl 23, id 29919, len 124)
```

Programy s podporou IPv6

Současné distribuce už obsahují klienty a servery s tolik potřebnou aktivovanou podporou IPv6. Viz nejdříve *http://www.bieringer.de/linux/IPv6/status/IPv6+Linux-status-distributions.html.* Pokud zde nic nenajdete, zkuste se ještě podívat na *http://www.bieringer.de/linux/IPv6/status/IPv6+Linux -status-apps.html*, zda byl program už převeden do tvaru IPv6 pod Linuxem. Některé rady k nej-častěji používaným programům naleznete i na adresách *http://www.bieringer.de/linux/IPv6/IPv6-HOWTO/IPv6-HOWTO-3.html* a *http://www.bieringer.de/linux/IPv6/IPv6-HOWTO/IPv6-HOWTO-4.html.*

Klientské programy s podporou IPv6 (výběr)

K provozování následujících testů je nutno mít systém aktivovaný pro IPv6 a některé příklady používají adresy, existuje-li spojení na 6bone.

Test DNS na rozlišování adres IPv6

Kvůli bezpečnostním aktualizacím v minulých letech by měl být na všech serverech DNS provo-zován novější software, který už rozumí (přechodnému) adresovému typu AAAA (novější jménem A6 ještě není v tomto okamžiku k dispozici, neboť ten musí mít podporu BIND9 a novějších a také dosud neexistující podporu kořenové domény IP6.ARPA). Jednoduchý test, zda systém umí zpra-covávat adresy IPv6, můžeme provést takto:

```
# host -t AAAA www.join.uni-muenster.de
```

a mělo by se ukázat něco takového:

```
www.join.uni-muenster.de. is an alias for tolot.join.uni-muenster.de. tolot.join.uni-muenster.de. has AAAA address
2001:638:500:101:2e0:81ff:fe24:37c6
```

Klienti telnet s podporou IPv6

K dispozici jsou klienti telnet s podporou IPv6. Jednoduchý test lze provést takto:

```
$ telnet 3ffe:400:100::1 80 Trying 3ffe:400:100::1... Connected to 3ffe:400:100::1. Escape character is '^]'. HEAD / HTTP/1.0 HTTP/1.1 200 OK
Date: Sun, 16 Dec 2001 16:07:21 GMT Server: Apache/2.0.28 (Unix) Last-Modified: Wed, 01 Aug 2001 21:34:42 GMT ETag: "3f02-a4d-
b1b3e080" Accept-Ranges: bytes Content-Length: 2637 Connection: close Content-Type: text/html; charset=ISO-8859-1 Connection closed by
foreign host.
```

Jestliže klient telnet nerozumí adrese IPv6 a řekne něco jako „cannot resolve hostname", IPv6 není aktualizovaná.

Klienti ssh s podporou IPv6

openssh

Stávající verze openssh podporují IPv6. V závislosti na konfiguraci před kompilací existuje ve dvou verzích:

bez implicitní ipv4: klient nejprve zkouší spojení IPv6, a když neuspěje, vrátí se k IPv4.
s implicitní ipv4: implicitní spojení je IPv4, spojení IPv6 musí být nastaveno, viz následují-cí příklad:

```
$ ssh -6 ::1 user@::1's password: ******  [user@ipv6host user]$
```

Když váš klient nerozumí volbě „-6“, nemá aktivovanou podporu IPv6, což platí pro většinu balí-ků SSH ve verzi 1.

ssh.com

Klient a server SSH z ssh.com také umí zpracovávat adresy IPv6 a je zdarma pro všechny systémy Linux a FreeBSD bez ohledu na soukromé či komerční využití.

Internetové prohlížeče s podporou IPv6

Momentální stav internetových prohlížečů s podporou IPv6 je k dispozici na *http://www.bieringer.de/linux/IPv6/status/IPv6+Linux-status-apps.html#HTTP*.

Většina z nich ovšem ještě dosud nemá vyřešené některé problémy:

Při používání proxy serveru, který je nastaven jen na IPv4, nebude tento server rozumět požadavku s IPv6 a odmítne jej. Řešení: aktualizace softwaru proxy serveru (viz dále).
Automatická nastavení proxy (*.pac) nelze rozšířit tak, aby zpracovával požadavky IPv6 odlišně (tedy nevyužíval proxy) kvůli jejich podstatě (jsou napsány v Java skriptu a zdro-jové texty jsou pevně zakódovány podobně jako ve zdrojovém kódu Maxilla).

Ani starší verze nerozumějí URL se zakódovanou adresou IPv6, jako např. *http:///[3ffe:400:100::1]/*(Toto URL pracuje pouze s prohlížečem s podporou IPv6!).Kterýkoli prohlížeč můžete otestovat pomocí uvedeného URL bez použití proxy.

Testovací URL

Brouzdání po Internetu s využitím adres IPv6 je vhodné začít na *http://www.kame.net/*. Pokud je želva na těchto stránkách animovaná, spojení je uskutečněno prostřednictvím IPv6, jinak se želva nehýbe.

Serverové programy s podporou IPv6

Tato část návodu se týká spíše klientů. Proto jsou informace o serverech s podporou IPv6, např. sshd, httpd, telnetd atd., popsány v kapitole „Popis démonů s aktivovanými IPv6“.

Často kladené otázky (Kontrola systému s podporou IPv6) Nástroje

Otázka: Programem ping6 se nemohu dostat na lokální adresy
Chybová zpráva: „*connect: Invalid argument*“ Jádro neví, jaké fyzické nebo virtuální spojení chcete použít pro odeslání těchto paketů ICMPv6.Proto se vypíše toto chybové hlášení.
Řešení: Uveďte např. rozhraní:

```
ping6 -I eth0 fe80::2e0:18ff:fe90:9205
```

Viz také kapitola „ping IPv6“.

Otázka: Jako normální uživatel se nemohu dostat k programům ping6 a traceroute6

Chybová zpráva: *„icmp socket: Operation not permitted“*Tyto nástroje vytvářejí speciální pakety ICMPv6 a odesílají je. To se provádí přímým využitím soke-
tů v jádru. Sokety však může přímo využívat pouze uživatel root. Proto se normálnímu uživateli
vypisuje tato zpráva.
Řešení: Pokud je opravdu nutné, aby směli tyto nástroje používat všichni uživatelé, můžete pomo-

cí `chmod u+s /path/to/program` přidat suid bit, viz kapitola „ping Ipv6“. Není-li to nutné, může-te změnit skupinu programu například na „wheel“, zařadit do této skupiny ty, kteří mají mít právopoužívat tento program, a ostatním uživatelům pomocí `chmod o-rwx /path/to/program` zaká-zat spuštění programu. Anebo zkonfigurujte sudo tak, aby se aktivovala vaše bezpečnostní poli-tika.

Konfigurace síových zařízení

Různá síová zařízení

V daném uzlu existují různá síťová zařízení. Lze je seskupit do tříd:

Připojené fyzicky, např. eth0, tr0.
Existující virtuálně, např. ppp0, tun0, tap0, sit0, isdn0, ippp0.

Fyzicky připojená

Fyzicky připojená rozhraní jako Ethernet nebo Token-Ring jsou normální zařízení a nevyžadují zvláštní zacházení.

Virtuálně připojená

Virtuálně připojená rozhraní potřebují vždy zvláštní podporu.

Tunelová rozhraní IPv6-in-IPv4

Tato rozhraní se obvykle nazývají sit*x*. Jméno *sit* je zkratka vzniklá ze slov Simple Internet Tran-sition. Toto zařízení například umí zapouzdřovat pakety IPv6 do paketů IPv4 a posílat je tunelem do jiného koncového bodu.

Rozhraní sit0 má speciální význam a nelze je používat pro tunely.

Rozhraní PPP

Rozhraní PPP poskytuje podporu IPv6 démon PPP s aktivovanou podporou IPv6.

Rozhraní ISDN HDLC

Podpora IPv6 pro HDLC se zapouzdřeným ip je zabudovaná v jádru.

Rozhraní ISDN PPP

V rozhraních ISDN PPP (ippp) není podpora IPv6 aktivovaná jádrem a ani není známo, že by ji měl někdo v plánu vytvořit, neboť v jádru 2.5+ budou nahrazeny obecnější vrstvou rozhraní ppp.
SLIP + PLIP

Jak jsme se už zmínili, tato rozhraní nepodporují přenos IPv6 (odesílání je možné, avšak určování příjemce nefunguje).

Ether-tap

Zařízení Ether-tap (zařízení pro monitorování sítě) podporují IPv6 a také jsou bezstavově zkonfi-gurována. Před používáním je nutno zavést modul ethertap.

Zařízení tun

Dosud jsem netestoval.

ATM

01/2002: Dosud nejsou podporovány jádrem vanilla, zatímco rozšířeními USAGI ano.

Ostatní

Zapomněl jsem na nějaké rozhraní?

Zapínání a vypínání rozhraní

Rozhraní můžeme zapínat a vypínat dvěma způsoby.

Příkaz ip

Použití:

ip link set dev <interface> up # ip link set dev <interface> down

Příklad:

ip link set dev eth0 up # ip link set dev eth0 down

Příkaz ifconfig

Použití:

/sbin/ifconfig <interface> up # /sbin/ifconfig <interface> down

Příklad:

/sbin/ifconfig eth0 up # /sbin/ifconfig eth0 down

Konfigurace adres IPv6

Adresu IPv6 můžete na rozhraní konfigurovat různými způsoby. Můžete použít příkazy ifconfig nebo ip.

Výpis existujících adres IPv6

Nejdříve byste si měli ověřit, zda a které adresy IPv6 jsou už zkonfigurované (pravděpodobně automaticky v průběhu bezstavové autokonfigurace).
Příkaz ip

Použití:

```
# /sbin/ip -6 addr show dev <interface>
```

Příklad staticky zkonfigurovaného počítače:

```
# /sbin/ip -6 addr show dev eth0
2: eth0: <BROADCAST,MULTICAST,UP&gt; mtu 1500 qdisc pfifo_ fast qlen 100 inet6 fe80::210:a4ff:fee3:9566/10 scope link inet6 3ffe:ffff:
0:f101::1/64 scope global inet6 fec0:0:0:f101::1/64 scope site
```

Příklad auto-konfigurovaného počítače:

Zde vidíte některé automaticky zkonfigurované adresy IPv6 a jejich životnost.

```
# /sbin/ip -6 addr show dev eth0
3: eth0: <BROADCAST,MULTICAST,PROMISC,UP&gt; mtu 1500 qdisc pfifo_fast qlen
  100 inet6 2002:d950:f5f8:f101:2e0:18ff:fe90:9205/64 scope global dynamic valid_lft 16sec preferred_lft 6sec inet6 3ffe:
400:100:f101:2e0:18ff:fe90:9205/64 scope global dynamic valid_lft 2591997sec preferred_lft 604797sec inet6 fe80::2e0:18ff:fe90:9205/10
  scope link
```

Příkaz ifconfig

Použití:

```
# /sbin/ifconfig <interface>
```

Příklad (výstup je odfiltrován pomocí programu grep tak, aby vypsal pouze adresu IPv6): Zde vidíte různé adresy IPv6 s různými poli působnosti.

```
# /sbin/ifconfig eth0 |grep “inet6 addr:” inet6 addr: fe80::210:a4ff:fee3:9566/10 Scope:Link inet6
addr: 3ffe:ffff:0:f101::1/64 Scope:Global inet6 addr: fec0:0:0:f101::1/64 Scope:Site
```

Přidání adresy IPv6

Přidání adresy IPv6 je podobné adresovému mechanismu „IP ALIAS“ při adresování rozhraní IPv4.

Příkaz ip

Použití:

```
# /sbin/ip -6 addr add <ipv6address>/<prefixlength> dev <interface>
```

Příklad:

```
# /sbin/ip -6 addr add 3ffe:ffff:0:f101::1/64 dev eth0
```

Příkaz ifconfig

Použití:

```
# /sbin/ifconfig <interface> inet6 add <ipv6address>/<prefixlength>
```

Příklad:

```
# /sbin/ifconfig eth0 inet6 add 3ffe:ffff:0:f101::1/64
```

Odstranění adresy IPv6

Nepoužívá se často. Pozor při odstraňování neexistující adresy IPv6, u některých starších jader to vede k havárii systému!

Příkaz ip

Použití:

```
# /sbin/ip -6 addr del <ipv6address>/<prefixlength> dev <interface>
```

Příklad:

```
# /sbin/ip -6 addr del 3ffe:ffff:0:f101::1/64 dev eth0
```

Příkaz ifconfig

Použití:

```
# /sbin/ifconfig <interface> inet6 del <ipv6address>/<prefixlength>
```

Příklad:

```
# /sbin/ifconfig eth0 inet6 del 3ffe:ffff:0:f101::1/64
```

Konfigurace normální trasy IPv6

Chcete-li opustit svoje spojení a posílat pakety do světa Internetu IPv6, je nutno provést směro-vání. Pokud už máte na svém připojení směrovač s aktivovanými adresami IPv6, stačí přidat trasu IPv6.

Výpis existujících tras IPv6

Nejdříve byste si měli ověřit, zda a které adresy IPv6 jsou už zkonfigurované (pravděpodobně automaticky prostou autokonfigurací).

Příkaz ip

Použití:

```
# /sbin/ip -6 route show [dev <device>]
```

Příklad:

```
# /sbin/ip -6 route show dev eth0 3ffe:ffff:0:f101::/64 proto kernel metric 256 mtu 1500 advmss 1440

fe80::/10 ff00::/8          proto kernel metric 256 mtu 1500 advmss 1440 proto kernel metric 256 mtu 1500 advmss
default                     1440 proto kernel metric 256 mtu 1500 advmss 1440
```

Příkaz route

Použití:

```
# /sbin/route -A inet6
```

Příklad (výstup je odfiltrován na rozhraní eth0). Na jednom rozhraní vidíte různé trasy IPV6 pro různé adresy.

```
# /sbin/route -A inet6 |grep -w "eth0" 3ffe:ffff:0:f101 ::/64 :: UA 256 0 0 eth0 <- Interface
route for global address fe80::/10 :: UA 256 0 0 eth0 <- Interface route for link-local
address ff00::/8 :: UA 256 0 0 eth0 <- Interface route for all multicast addresses ::/0 ::
UDA 256 0 0 eth0 <- Automatic default route
```

Přidání trasy IPv6 pomocí brány

Většinou potřebné k dosažení vnější sítě s adresou IPv6 pomocí směrovače s aktivací IPv6 na spojení.

Příkaz ip

Použití:

```
# /sbin/ip -6 route add <ipv6network>/<prefixlength> via <ipv6address> ¬ [dev <device>]
```

Příklad:

```
# /sbin/ip -6 route add 2000::/3 via 3ffe:ffff:0:f101::1
```

Příkaz route

Použití:

```
# /sbin/route -A inet6 add <ipv6network>/<prefixlength> gw ¬ <ipv6address> [dev <device>]
```

Zařízení může být potřebné také tehdy, je-li adresa IPv6 brány lokální.
V následujícím příkladu je trasa přidána ke všem globálním adresám (2000::/3) pomocí brány 3ffe:ffff:0:f101::1.

```
# /sbin/route -A inet6 add 2000::/3 gw 3ffe:ffff:0:f101::1
```

Odstranění trasy IPv6 pomocí brány

Ručně se provádí zřídka, většinou síťovými konfiguračními skripty při vypínání systému (plném nebo po jednotlivých rozhraních).

Příkaz ip

Použití:

/sbin/ip -6 route del <ipv6network>/<prefixlength> via <ipv6address> ¬ [dev <device>]

Příklad:

/sbin/ip -6 route del 2000::/3 via 3ffe:ffff:0:f101::1

Příkaz route

Použití:

/sbin/route -A inet6 del <network>/<prefixlength> [dev <device>]

Příklad na odstranění shora přidané trasy:

/sbin/route -A inet6 del 2000::/3 gw 3ffe:ffff:0:f101::1

Přidání trasy IPv6 pomocí rozhraní

Používá se zřídka, většinou jen v případě jednoúčelového dvoubodového spojení.

Příkaz ip

Použití:

/sbin/ip -6 route add <ipv6network>/<prefixlength> dev <device> ¬ metric 1

Příklad:

/sbin/ip -6 route add 2000::/3 dev eth0 metric 1

Metrika 1 je zde použita kvůli kompatibilitě s metrikou v trase, neboť implicitní metrika v příkazu ip je 1024.

Příkaz route

Použití:

/sbin/route -A inet6 add <network>/<prefixlength> dev <device>

Příklad:

/sbin/route -A inet6 add 2000::/3 dev eth0

Odstranění trasy IPv6 pomocí rozhraní

Ručně se provádí zřídka, většinou síťovými konfiguračními skripty při vypínání systému.

Příkaz ip

Použití:

/sbin/ip -6 route del <ipv6network>/<prefixlength> dev <device>

Příklad:

/sbin/ip -6 route del 2000::/3 dev eth0

Příkaz route

Použití:

/sbin/route -A inet6 del <network>/<prefixlength> dev <device>

Příklad:

/sbin/route -A inet6 del 2000::/3 dev eth0

Často kladené otázky k trasám IPv6 Podpora implicitní trasy IPv6

Jednou z idejí IPv6 je hierarchické směrování, proto je ve směrovačích nutných méně položek. V současných jádrech Linuxu jsou určité problémy:

Klienti (kteří nesměrují pakety!)

Klient může nastavit implicitní trasu pomocí prefixu „::/0“, toto směrování mohou zadat i při auto-konfiguraci například pomocí radvd na spojení, jak je zřejmé z následujícího příkladu:

ip -6 route show | grep ^default default via fe80::212:34ff:fe12:3450 dev eth0 proto kernel metric 1024
expires 29sec mtu 1500 advmss 1440

Přeposílání paketů směrovači

Současná jádra Linuxu (přinejmenším <= 2.4.17) nepodporují implicitní trasy. Můžete je nastavit, avšak vyhledávání trasy zhavaruje v okamžiku, kdy by měl být paket přeposlán (což je běžná činnost směrovače).

Z toho důvodu lze v současnosti nastavit implicitní směrování pouze pomocí prefixu globální adresy 2000::/3.

> Poznámka
>
> Opatrně při implicitním směrování bez filtrování adresy na krajních směrovačích. Jinak se vícesměrový přenos nebo přeprava v místní síti dostane ven.

Zjišování sousedů

Zjišťování sousedů je následníkem protokolu ARP (Address Resolution Protocol) v IPv4. Můžete získávat informace o svých sousedech a navíc můžete tyto položky nastavovat a rušit. Jádro sle-duje úspěšnou detekci souseda (jako ARP v IPv4). Do tabulky sousedů můžete zasahovat pomo-cí příkazu ip.

Výpis sousedů pomocí ip

Zjištěné a zkonfigurované sousedy IPv6 můžete vypsat následujícím příkazem:

```
# ip -6 neigh show [dev <device>]
```

V následujícím příkladu vidíme jednoho souseda, kterým je směrovač v dosahu:

```
# ip -6 neigh show fe80::201:23ff:fe45:6789 dev eth0 lladdr 00:01:23:45:67:89 router nud reachable
```

Provádění změn v tabulce sousedů příkazem ip Ruční přidání položky

Následujícím příkazem můžete přidat do tabulky položku:

```
# ip -6 neigh add <IPv6 address> lladdr <link-layer address> dev <device>
```

Příklad:

```
# ip -6 neigh add fec0::1 lladdr 02:01:02:03:04:05 dev eth0
```

Ruční zrušení položky

Položku lze do tabulky nejen přidat, nýbrž i odstranit:

```
# ip -6 neigh del <IPv6 address> lladdr <link-layer address> dev <device>
```

Příklad:

```
# ip -6 neigh del fec0::1 lladdr 02:01:02:03:04:05 dev eth0
```

Pokročilejší nastavení

Nástroj ip není příliš dobře zdokumentován, avšak je velmi silný. Podrobnosti viz on-line nápo-věda:

```
# ip -6 neigh help
Usage: ip neigh { add | del | change | replace } { ADDR [ lladdr LLADDR ]
                  [ nud { permanent | noarp | stale | reachable } ]
                  | proxy ADDR } [ dev DEV ]
       ip neigh {show|flush} [ to PREFIX ] [ dev DEV ] [ nud STATE ]
```

Vypadá to, jako by některé volby byly platné jen pro IPv4 ... můžete-li přispět informacemi o vol-bách a pokročilejším použití, prosím, pošlete je.

Konfigurace tunelů IPv6-in-IPv4

Chcete-li opustit svoje připojení a kolem sebe nemáte síť s podporou IPv6, k dosažení Internetus IPv6 potřebujete tunel typu IPv6-in-IPv4.Existují určité druhy tunelových mechanismů a také některé možnosti nastavení.

Typy tunelů

Existuje několik možností, jak posílat pakety IPv6 přes připojení IPv4.

Statický dvoubodový tunel: 6bone

Dvoubodový tunel je jednoúčelový tunel do koncového bodu, který ví o síti IPv6 (kvůli zpětné-mu směrování) a zná adresu IPv4 koncového bodu tunelu. Je definován v RFC 2893. Požadavky:

Adresa IPv4 lokálního koncového bodu tunelu musí být statická, globálně jednoznačná a dosažitelná z koncového bodu cizího tunelu.
Musíte mít přiřazen prefix IPv6 (viz registr 6bone).
Koncový bod cizího tunelu, který je schopen nasměrovat váš prefix IPv6 na váš lokální koncový bod tunelu (většinou vyžaduje vzdálenou ruční konfiguraci).

Automatické vytváření tunelů

K automatickému vytvoření tunelu dojde, když je některý uzel přímo napojen na jiný uzel, jehož adresu IPv4 získal už dříve.

Vytváření tunelů 6to4

Vytváření tunelů 6to4 (RFC 3056) využívá jednoduchého mechanismu. Každý uzel s globálně jednoznačnou adresou IPv4 může být koncovým bodem tunelu 6to4 (není-li provoz blokován firewallem). Vytváření tunelů 6to4 většinou není obousměrné. Tento případ lze rozdělit na tunelování jedním nebo druhým směrem. Speciální adresa IPv6 tedy určuje, že tento uzel bude pro napojení na globální síť IPv6 využívat tunel 6to4.
Generování prefixu 6to4

Adresa 6to4 je definovaná takto (schéma je převzato z RFC 3056):

| 3+13 | 32 | 16 | 64 bits | +---+------+-----------+--------+---------------------------------+ | FP+TLA | V4ADDR | SLA ID | ID rozhraní | | 0x2002 | | | | +---+------+-----------+--------+---------------------------------+

FP a TLA mají dohromady (16 bitů) hodnotu 0x2002. V4ADDR je jednoznačná adresa IPv4 uzlu (v hexadecimálním tvaru). SLA je identifikátor podsítě (může existovat 65 536 podsítí) a lze jej použít pro lokální strukturu sítí.

V případě brány je tento prefix generován pomocí SLA „0000“ a přípony „::1“ (nikoli nutně, stačí, když je lokální) a přiřadí se rozhraní tunelu 6to4. Poznamenejme, že Microsoft Windows používají V4ADDR také jako příponu.

Tunel 6to4 směrem ven

Uzel musí vědět, kterému koncovému bodu cizího tunelu má poslat pakety IPv6 zabalené do paketů IPv4. V začátcích tunelů 6to4 byly definovány jednoúčelové přijímací směrovače pro ode-sílání ven. Seznam směrovačů viz *http://www.kfu.com/~nsayer/6to4/*.

Nyní lze odesílací směrovače 6to4 nalézt automaticky pomocí výběrové adresy 192.88.99.1, kterou směrovací protokoly zpracují na pozadí, podrobnosti viz RFC 3068.

Tunel 6to4 směrem dovnitř

Tunel směrem dovnitř (6bone -> váš uzel aktivovaný pro 6to4) není pevný a může se pro různé cizí počítače lišit v závislosti na tom, komu byly původní pakety odeslány. Existují dvě možnosti:

Cizí počítač použije 6to4 a pošle paket přímo zpět na váš uzel (viz níže).
Cizí počítač posílá pakety zpět do globální sítě IPv6 a předávací směrovač vytváří automa-tický tunel zpět na váš uzel v závislosti na dynamickém směrování.

Možnosti provozu 6to4

Z 6to4 do 6to4: normální přímý tunel mezi dvěma počítači s podporou 6to4.
Z 6to4 do non-6to4: posílá se tunelem směrem ven.
Z non-6to4 do 6to4: posílá se tunelem směrem dovnitř.

Výpis existujících tunelů Příkaz ip

Použití:

```
# /sbin/ip -6 tunnel show [<device>]
```

Příklad:

```
# /sbin/ip -6 tunnel show sit0: ipv6/ip remote any local any ttl 64 nopmtudisc sit1: ipv6/ip remote 195.226.187.50 local any ttl 64
```

Příkaz route

Použití:

```
# /sbin/route -A inet6
```

Příklad (výstup je filtrován tak, aby se vypisovaly pouze tunely vedoucí virtuálním rozhraním sit0):

```
# /sbin/route -A inet6 | grep "\Wsit0\W*$" ::/96 :: U 256 2 0 sit0 2002::/16 :: UA 256 0 0 sit0
2000::/3 ::193.113.58.75 UG 1 0 0 sit0 fe80::/10 :: UA 256 0 0 sit0 ff00::/8 :: UA 256 0 0
sit0
```

Nastavení dvoubodových tunelů

Dvoubodové tunely můžeme přidávat nebo rušit třemi různými způsoby.Užitečné doplňující informace o nastavování tunelů příkazem ip naleznete na adrese

http://www.deepspace6.net/docs/iproute2tunnel-en.html.

Přidání dvoubodového tunelu

Příkaz ip

Metoda používaná pro přidávání malého počtu tunelů.
Použití pro vytvoření tunelového zařízení (které však potom nezůstane zapnuté, a musí být uve
deno také TTL, neboť implicitní hodnota je 0):

```
# /sbin/ip tunnel add <device> mode sit ttl <ttldefault> remote ¬ <ipv4addressofforeigntunnel> local <ipv4addresslocal>
```

Použití (obecný příklad pro tři tunely):

```
# /sbin/ip tunnel add sit1 mode sit ttl <ttldefault> remote ¬ <ipv4addressofforeigntunnel1> local <ipv4addresslocal> # /sbin/ip link set dev sit1 up # /sbin/ip -6 route add <prefixtoroute1> dev sit1 metric 1 # /sbin/ip tunnel add sit2 mode sit ttl <ttldefault> ¬ <ipv4addressofforeigntunnel2> local <ipv4addresslocal> # /sbin/ip link set dev sit2 up # /sbin/ip -6 route add <prefixtoroute2> dev sit2 metric 1 # /sbin/ip tunnel add sit3 mode sit ttl <ttldefault> ¬ <ipv4addressofforeigntunnel3> local <ipv4addresslocal> # /sbin/ip link set dev sit3 up # /sbin/ip -6 route add <prefixtoroute3> dev sit3 metric 1
```

Příkazy ifconfig a route (neschválené)

Tento způsob přidávání tunelů nelze příliš doporučit, neboť je poněkud podivný. Přidáváte-li jeden tunel, problém nenastane, avšak nastavíte-li více než jeden, první nejdou zavřít tak, aby ostatní zůstaly v činnosti.

Použití (obecný příklad pro tři tunely):

```
# /sbin/ifconfig sit0 up # /sbin/ifconfig sit0 tunnel <ipv4addressofforeigntunnel1> # /sbin/ifconfig sit1 up # /sbin/route -A inet6 add <prefixtoroute1> dev sit1 # /sbin/ifconfig sit0 tunnel <ipv4addressofforeigntunnel2> # /sbin/ifconfig sit2 up # /sbin/route -A inet6 add <prefixtoroute2> dev sit2 # /sbin/ifconfig sit0 tunnel <ipv4addressofforeigntunnel3> # /sbin/ifconfig sit3 up # /sbin/route -A inet6 add <prefixtoroute3> dev sit3
```

> Důležité
>
> Tyto příkazy NEPOUŽÍVEJTE, neboť toto nastavení implicitně aktivuje „automatické tunelování“ odkudkoli z Internetu, což je neobhajitelné riziko.

Pouze příkaz route

Tunely také můžete nastavit způsobem Non Broadcast Multiple Access (NBMA), což je jednoduchý způsob, jak přidat mnoho tunelů současně. Žádný z nich však nemůže být číslován (což ovšem není požadovaná vlastnost).

Použití (obecný příklad pro tři tunely):

```
# /sbin/ifconfig sit0 up # /sbin/route -A inet6 add <prefixtoroute1> gw ¬ ::<ipv4addressofforeigntunnel1> dev sit0 # /sbin/route -A inet6 add <prefixtoroute2> gw ¬ ::<ipv4addressofforeigntunnel2> dev sit0 # /sbin/route -A inet6 add <prefixtoroute3> gw ¬ ::<ipv4addressofforeigntunnel3> dev sit0
```

Důležité: Tyto příkazy NEPOUŽÍVEJTE, neboť toto nastavení implicitně aktivuje „automatické tunelování“ odkudkoli z Internetu, což je neobhajitelné riziko.

Odstraňování dvoubodových tunelů

Manuálně se tento způsob nepoužívá často, avšak používají jej skripty pro čisté ukončení nebo restart konfigurace IPv6.

Příkaz ip

Použití pro odstranění tunelového zařízení:

```
# /sbin/ip tunnel del <device>
```

Použití (obecný příklad pro tři tunely):

```
# /sbin/ip -6 route del <prefixtoroute1> dev sit1 # /sbin/ip link set sit1 down # /sbin/ip tunnel del sit1 # /sbin/ip -6 route del <prefixtoroute2> dev sit2 # /sbin/ip link set sit2 down # /sbin/ip tunnel del sit2 # /sbin/ip -6 route del <prefixtoroute3> dev sit3 # /sbin/ip link set sit3 down # /sbin/ip tunnel del sit3
```

Příkazy ifconfig a route (neschválený, protože jejich použití není žádná legrace)

Nejen že vytváření je podivné, ale podivné je i ukončení ... tunely musíte odstraňovat v obráce-ném pořadí, což znamená, že posledně vytvořený musí být odstraněn jako první. Použití (obecný příklad pro tři tunely):

/sbin/route -A inet6 del <prefixtoroute3> dev sit3 # /sbin/ifconfig sit3 down # /sbin/route -A inet6 del <prefixtoroute2> dev sit2 # /sbin/ifconfig sit2 down # /sbin/route -A inet6 add <prefixtoroute1> dev sit1 # /sbin/ifconfig sit1 down # /sbin/ifconfig sit0 down

Příkaz „route"

Stejně se odstraňují normální trasy IPv6. Použití (obecný příklad

pro tři tunely):

/sbin/route -A inet6 del <prefixtoroute1> gw ¬ ::<ipv4addressofforeigntunnel1> dev sit0
/sbin/route -A inet6 del <prefixtoroute2> gw ¬ ::<ipv4addressofforeigntunnel2> dev sit0
/sbin/route -A inet6 del <prefixtoroute3> gw ¬ ::<ipv4addressofforeigntunnel3> dev sit0
/sbin/ifconfig sit0 down

Číslované dvoubodové tunely

Někdy potřebujete zkonfigurovat dvoubodové tunely s adresami IPv6 jako v případě IPv4. To je možné pouze s prvním (ifconfig +route – zamítnutým) a třetím (ip+route) nastavením tunelů. V takových případech můžete k rozhraní tunelu přidat adresy IPv6, jak je zřejmé z konfigurace rozhraní.

Nastavení tunelů 6to4

Všimněte si, že podpora tunelů 6to4 v jádru vanilla řady 2.2.x chybí (další informace viz kapitola „Jádro a IPv6"). Také si všimněte, že délka prefixu pro adresu 6to4 je kvůli sítím 16, všechny ostatní počítače aktivované pro 6to4 jsou na téže vrstvě 2.

Přidání tunelu 6to4

Nejprve musíte vypočítat prefix 6to4 z lokálně přiřazené globálně směrovatelné adresy IPv4 (nemá-li váš počítač žádnou globálně směrovatelnou adresu IPv4, ve zvláštních případech je na hraničních bránách možný NAT):

Předpokládejme, že vaše IPv4 adresa je:

1.2.3.4

vygenerovaný prefix 6to4 bude:

2002:0102:0304::

Lokální brány 6to4 by vždy měly mít (není to však nutné, můžete zvolit libovolnou lokální příponu, bude-li vám to vyhovovat) přiřazenou příponu „::1", proto bude vaše lokální adresa 6to4:

2002:0102:0304::1

Pro automatickou generaci můžete použít například:

ipv4="1.2.3.4"; printf "2002:%02x%02x:%02x%02x::1" `echo $ipv4 | tr "." " "`

Nyní tedy existují dva způsoby nastavení tunelů 6to4.

Příkaz ip a jednoúčelové tunelové zařízení

V současné době je to doporučovaný způsob (je nutno specifikovat TTL, protože implicitní hodnota je 0).

Vytvoření nového tunelového zařízení:

/sbin/ip tunnel add tun6to4 mode sit ttl <ttldefault> remote any local <localipv4address>

Zapojte rozhraní:

/sbin/ip link set dev tun6to4 up

K rozhraní přidejte lokální adresu 6to4 (Poznámka: Délka prefixu 16 je důležitá!):

/sbin/ip -6 addr add <local6to4address>/16 dev tun6to4

Ke globální síti IPv6 přidejte pomocí výběrové adresy IPv4 všech směrovačů 6to4 (implicitní) trasu:

/sbin/ip -6 route add 2000::/3 via ::192.88.99.1 dev tun6to4 metric 1

Bylo oznámeno, že některé verze ip (např. v SuSE Linuxu 9.0) nepodporují adresy IPv6 kompa-tibilní s IPv4 pro brány, v takovém případě musí být použita odpovídající adresa IPv6:

/sbin/ip -6 route add 2000::/3 via 2002:c058:6301::1 dev tun6to4 metric 1

Příkazy ifconfig a route a obecné tunelové zařízení „sit0" (neschváleno)

Toto nyní není schváleno, protože používání obecného tunelu zařízení sit0 neumožňuje filtraci po jednotlivých zřízeních.
Zapojení obecného tunelového rozhraní sit0:

```
# /sbin/ifconfig sit0 up
```

K rozhraní přidejte lokální adresu 6to4:

```
# /sbin/ifconfig sit0 add <local6to4address>/16
```

Ke globální síti IPv6 přidejte pomocí výběrové adresy IPv4 na všechna předání 6to4 (implicitní) trasu:

```
# /sbin/route -A inet6 add 2000::/3 gw ::192.88.99.1 dev sit0
```

Odstranění tunelu 6to4

Příkaz ip a jednoúčelové tunelové zařízení

Odstranění všech tras pomocí tohoto jednoúčelového tunelového zařízení:

```
# /sbin/ip -6 route flush dev tun6to4
```

Zavřete rozhraní:

```
# /sbin/ip link set dev tun6to4 down
```

Odstraňte vytvořená tunelová zařízení:

```
# /sbin/ip tunnel del tun6to4
```

Příkazy ifconfig a route a obecné tunelové zařízení „sit0“ (neschváleno)

Odstraňte (implicitní) trasu pomocí tunelového rozhraní 6to4:

```
# /sbin/route -A inet6 del 2000::/3 gw ::192.88.99.1 dev sit0
```

Odstraňte lokální adresu 6to4 na rozhraní:

```
# /sbin/ifconfig sit0 del <local6to4address>/16
```

Vypněte obecné tunelové zařízení (s tím opatrně, možná se ještě používá...)

```
# /sbin/ifconfig sit0 down
```

> Poznámka
>
> Kapitola o konfiguraci tunelů IPv4-in-IPv6 bude doplněna později. V daném okamžiku se tyto tunely používají spíše v testovacím prostředí; vypadá to, že v současné době chybí podpora pro Linux (03/2004). Informace mezitím získáte v RFC 2473.

Nastavení jádra v souborovém systému /proc

> Poznámka
>
> Převážná část této kapitoly pochází ze souboru ip-sysctl.txt, který je součástí stávající-ho zdrojového kódu v adresáři Documentation/networking. Díky Pekku Savolovi za údržbu té části souboru, která se týká IPv6. Text této kapitoly je víceméně zkopírován z tohoto dokumentu.

Jak zacházet se souborovým systémem /proc Příkazy cat a echo

Nejjednodušší přístup do souborového systému /proc vám umožní příkazy cat a echo, musí však splňovat určité požadavky:

- Souborový systém /proc musí být aktivován v jádru, tj. při překladu musí být nastaven tento přepínač:
  CONFIG_PROC_FS=y
- Souborový systém /proc je nutno nejdříve připojit, což můžeme otestovat takto:
  # mount | grep “type proc” none on /proc type proc (rw)
- K souborovému systému /proc musíte mít přístupová práva ke čtení, někdy i k zápisu (což obvykle má pouze uživatel root).

Obvykle se zapisuje pouze do položek v /proc/sys/*, z ostatních je možno jen číst a slouží pouze k získávání informací.

Přečtení hodnoty

Hodnotu položky můžete přečíst pomocí cat:

cat /proc/sys/net/ipv6/conf/all/forwarding 0

Nastavení hodnoty

Novou hodnotu můžete nastavit (je-li možno do položky zapisovat) pomocí příkazu echo:

echo “1” >/proc/sys/net/ipv6/conf/all/forwarding

Příkaz sysctl

Program sysctl zajišťuje přístup k přepínačům jádra a patří k moderním nástrojům. Lze jej použít i tehdy, když souborový systém není připojen. Máte však přístup pouze k /proc/sys/*! Program sysctl je součástí balíku procps (v systému Red Hat Linux).

- Rozhraní sysctl musí být aktivováno v jádru, tj. při kompilaci musel být zapnut přepínač:

CONFIG_SYSCTL=y

Čtení hodnoty

Nyní můžete přečíst hodnotu:

sysctl net.ipv6.conf.all.forwarding net.ipv6.conf.all.forwarding = 0

Nastavení hodnoty

Novou hodnotu lze nastavit (je-li možno do položky zapisovat):

sysctl -w net.ipv6.conf.all.forwarding=1 net.ipv6.conf.all.forwarding = 1

Poznámka: Při nastavování hodnoty nesmí být zleva ani zprava u rovnítka mezery. Je-li na řádku více hodnot, musí být uzávorkovány jako v následujícím příkazu:

sysctl -w net.ipv4.ip_local_port_range=”32768 61000” net.ipv4.ip_local_port_range = 32768 61000

Doplnění

Poznámka

V praxi existují i verze, které místo „.“ vypisují „/“.

Podrobnosti viz manuálové stránky sysctl.

Tip

Rychlý přehled o nastavení získáte volbou -a (výpis všech položek) ve spojení s příkazem grep.

Jaké najdete v souborovém systému /proc hodnoty

V souborovém systému /proc jsou hodnoty v několika formátech:

BOOLEAN: „0“ (nepravda) a „1“ (pravda).
INTEGER: Používá se i celočíselná hodnota.
Sofistikovanější řádky s několika hodnotami: Někdy se vypisuje také hlavička, pokud ne, informace o významu jednotlivých hodnot naleznete ve zdrojovém kódu jádra...

Položky v /proc/sys/net/ipv6/ conf/default/*

Změna implicitního nastavení určitého rozhraní.

conf/all/*

Změna nastavení všech rozhraní.Výjimka: conf/all/forwarding má zde jiný význam.

conf/all/forwarding

- Typ: BOOLEAN Aktivace globálního přeposílání IPv6 mezi všemi rozhraními. V IPv6 nemůžete řídit přeposílání na

každém rozhraní, řízení přeposílání se musí provádět pomo
cí množiny pravidel síťového filtru IPv6 (řízeného ip6tables) a je nutno specifikovat vstupní a výstupní zařízení (viz kapitola „Firewally s filtry netfilter6“). Tím se IPv6 liší od IPv4, kde je možno řídit přeposílání po jednotlivých zařízeních (rozhodnutí se provede na rozhraní, přes něž paket přišel).
Tím se také nastaví všechna rozhraní počítačů/směrovačů na „přeposílání“ na určitou hodnotu.

Podrobnosti viz dále. Tento způsob nazýváme globální přeposílání.
Je-li tato hodnota 0, není aktivováno přeposílání, pakety nikdy neopustí jiné rozhraní, a to ani

fyzické, ani logické (např. tunely).

conf/interface/*

Změna speciálního nastavení po jednotlivých rozhraních.
Funkce určitého nastavení je odlišná v závislosti na tom, zda je aktivováno lokální přeposílání či nikoli.

accept_ra

Typ: BOOLEAN.
Implicitní nastavení funkce: Aktivovaná, když je lokální přeposílání zablokované, zabloko

vaná, když je lokální přeposílání aktivované. Přijetí oznámení ze směrovače a autokonfigurace tohoto rozhraní se zaslanými daty.

accept_redirects

Typ: BOOLEAN.
Implicitní nastavení funkce: Aktivovaná, když je lokální přeposílání zablokované, zabloko

vaná, když je lokální přeposílání aktivované. Přijetí přesměrování zaslané směrovačem Ipv6.

autoconf

Typ: BOOLEAN.
Implicitní hodnota: TRUE.

Konfigurace lokální adresy (viz také kapitola „“) pomocí hardwarové adresy L2. Na rozhraní s fyzickou adresou L2 je automaticky vygenerovaná adresa, např. fe80::201:23ff:fe45:6789.

dad_transmits

Typ: INTEGER.
Implicitní hodnota: 1. Zaslání počtu zjištěných duplikátních adres.

forwarding

Typ: BOOLEAN.
Implicitní hodnota: FALSE, je-li zablokováno globální přeposílání, jinak TRUE. Konfigurace chování počítače/směrovače v závislosti na rozhraní.

Poznámka

Je vhodné mít na všech rozhraních stejné nastavení; smíšené scénáře nastavení směrovač/počítač jsou neobvyklé.

■ Hodnota FALSE: Implicitně se předpokládá chování počítače. To znamená:
1. V oznámení souseda není nastaven příznak IsRouter. Je-li to nutné, posílají se žádosti o směrovač.
Má-li accept_ra hodnotu TRUE (implicitní nastavení), přijmi oznámení směrovače (a proveď autokonfiguraci).
Má-li accept_redirects hodnotu TRUE (implicitní nastavení), přijmi přesměrování.
■ Hodnota TRUE: Je-li aktivováno lokální přeposílání, předpokládá se chování směrovače. Míní se tím pravý opak toho, co bylo uvedeno shora:
V oznámení souseda je nastaven příznak IsRouter.
Žádosti o směrovač se neposílají.
Oznámení směrovače jsou ignorována.
Přesměrování jsou ignorována.

hop_limit

Typ: INTEGER.
Implicitní hodnota: 64. Nastavení implicitní hodnoty Hop Limit.

mtu

Typ: INTEGER.

Implicitní hodnota: 1 280 (minimum vyžadované IPv6). Implicitní MTU.

router_solicitation_delay

Typ: INTEGER.
Implicitní hodnota: 1. Délka zpoždění v sekundách poté, co je rozhraní zapojeno před odesláním žádosti o směrovač.

router_solicitation_interval

Typ: INTEGER.
Implicitní hodnota: 4. Délka zpoždění v sekundách mezi dvěma odesláními žádosti o směrovač.

router_solicitations

Typ: INTEGER.
Implicitní hodnota: 3.

Počet žádostí o směrovač, které se pošlou, než se usoudí, že žádné směrovače nejsou přítomny.

neigh/default/*

Změna implicitního nastavení pro detekci sousedů a některé speciální globální intervaly a prahové hodnoty:

gc_thresh1

Typ: INTEGER.
Implicitní hodnota: 128.

gc_thresh2

Typ: INTEGER.
Implicitní hodnota: 512.

gc_thresh3

Typ: INTEGER.
Implicitní hodnota: 1 024. Ladicí parametr pro velikost tabulky sousedů. Máte-li mnoho rozhraní a problémy s počátky tras, chovají-li se podivně a havarují, tuto hodnotu

zvětšete. Anebo když běžící zebra (směrovací démon, *http://www.zebra.org/*) ohlásí:

ZEBRA: netlink-listen error: No buffer space available, type=RTM_NEWROUTE(24), seq=426, pid=0

gc_interval

Typ: INTEGER.
Implicitní hodnota: 30.

neigh/interface/*

Změna speciálního nastavení jednotlivých rozhraní kvůli detekci sousedů.

anycast_delay

Typ: INTEGER.
Implicitní hodnota: 100.

gc_stale_time

Typ: INTEGER.
Implicitní hodnota: 60.

proxy_qlen

Typ: INTEGER.
Implicitní hodnota: 64.

unres_qlen

Typ: INTEGER.
Implicitní hodnota: 3.

app_solicit

Typ: INTEGER.
Implicitní hodnota: 0.

locktime

Typ: INTEGER.
Implicitní hodnota: 0.

retrans_time

Typ: INTEGER.
Implicitní hodnota: 100.

base_reachable_time

Typ: INTEGER.
Implicitní hodnota: 30.

mcast_solicit

Typ: INTEGER.
Implicitní hodnota: 3.

ucast_solicit

Typ: INTEGER.
Implicitní hodnota: 3.

delay_first_probe_time

Typ: INTEGER.
Implicitní hodnota: 5.

proxy_delay

Typ: INTEGER.
Implicitní hodnota: 80.

route/*

Mění globální nastavení routování.

flush

Odstraněno v novějších jádrech.

gc_interval

Typ: INTEGER.
Implicitní hodnota: 30.

gc_thresh

Typ: INTEGER.
Implicitní hodnota: 1 024.

mtu_expires

Typ: INTEGER.
Implicitní hodnota: 600.

gc_elasticity

Typ: INTEGER.
Implicitní hodnota: 0.

gc_min_interval

Typ: INTEGER.
Implicitní hodnota: 5.

gc_timeout

Typ: INTEGER.

Implicitní hodnota: 60.

min_adv_mss

Typ: INTEGER.
Implicitní hodnota: 12.

max_size

Typ: INTEGER.
Implicitní hodnota: 4 096.

Položky IPv6 v /proc/sys/net/ipv4/

V tomto okamžiku (a bude tomu tak, dokud nebude IPv4 kompletně převeden do nezávislého modulu jádra) se pro IPv6 také používají některé přepínače.

ip_*

ip_local_port_range

Toto řídicí nastavení používá i IPv6.

tcp_*

Tato řídicí nastavení používá i IPv6.

icmp_*

Tato řídicí nastavení IPv6 nepoužívá. K aktivaci omezení rychlosti ICMPv6 (které lze velmi doporučit kvůli schopnosti čelit útokům ICMPv6) je nutno respektovat pravidla netfilter-v6.

Ostatní

V souvislosti s IPv6 nejsou známy.

Položky IPv6 v /proc/net/

V /proc/net existuje několik položek, které můžete pouze číst. Informace z nich nelze přečíst příkazem sysctl, musíte použít cat.

if_inet6

- Typ: Každé adrese odpovídá jeden řádek s několika hodnotami. Zde jsou uloženy všechny zkonfigurované adresy ve zvláštním tvaru. V příkladu je uvedeno pouze rozhraní loopback. Význam je popsán v dalším textu (další informace viz zdrojový kód jádra, sou-bor net/ipv6/addrconf.c).

```
# cat /proc/net/if_inet6 00000000000000000000000000000001 01 80 10 80 lo
+--------------------------------+ ++ ++ ++ ++ ++ | | | | | |
                                  2  3  4  5  6
```

Adresa IPv6 má 32 hexadecimálních znaků bez dvojteček.
Číslo síťového zařízení (index rozhraní) je v hexadecimálním tvaru (viz také „ip addr“).
Délka prefixu hexadecimálně.
Hodnota rozsahu platnosti (další viz include/net/ipv6.h a net/ipv6/addrconf.c ve zdrojovém kódu jádra).

5. Příznaky rozhraní (další viz include/linux/rtnetlink.h a net/ipv6/addrconf.c).
6. Jméno zařízení.

ipv6_route

- Typ: Každé trase odpovídá jeden řádek s několika hodnotami.

  Zde jsou uloženy všechny trasy IPv6 ve zvláštním tvaru. V příkladu je uvedeno pouze roz-hraní loopback. Význam je popsán v dalším textu (další informace viz net/ipv6/route.c). # cat /proc/net/ipv6_route 00000000000000000000000000000000 00 00000000000000000000000000000000 00 +--------------------------------+ ++ +--------------------------------+ ++ | | | | 1 2 3 4 ¬ 00000000000000000000000000000000 ffffffff 00000001 00000001 00200200 lo ¬ +--------------------------------+ +------+ +------+ +------+ +------+ ++ ¬ | | | | | | ¬ 5 6 7 8 9 10

Cílová síť IPv6 má 32 hexadecimálních znaků bez dvojteček.
Délka cílového prefixu IPv6 hexadecimálně.
Zdrojová síť IPv6 má 32 hexadecimálních znaků bez dvojteček.
Délka zdrojového prefixu IPv6 hexadecimálně.

Další směrovač IPv6, 32 hexadecimálních znaků bez dvojteček.
Metrika hexadecimálně.
Referenční čítač.
Uživatelský čítač.
Příznaky.
Jméno zařízení.

sockstat6

- Typ: Každému protokolu odpovídá jeden řádek s popisem a s hodnotou. Statistiky použitých soketů IPv6.

Příklad:

```
# cat /proc/net/sockstat6 TCP6: inuse 7 UDP6: inuse 2 RAW6: inuse 1 FRAG6: inuse 0 memory 0
```

snmp6

- Typ: Každému popisu SNMP a hodnotě odpovídá jeden řádek. Statistiky SNMP, lze je získat softwarem pro správu sítí ze serveru SNMP a příslušné tabulky MIB.

ip6_tables_names

Tabulky netfilter6, které jsou k dispozici.

Ladění v síti

Připojování serverových soketů.

Kontrola připojení serverových soketů příkazem netstat

Vědět, který serverový soket je aktivní na uzlu, je vždy zajímavé. Jednoduše můžete získat tyto

informace pomocí příkazu netstat.Volby: -nlptuPříklad:

```
# netstat -nlptu
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address PID/Program name
tcp 0 0 0.0.0.0:32768
¬ 1258/rpc.statd
tcp 0 0 0.0.0.0:32769
¬ 1502/rpc.mountd
tcp 0 0 0.0.0.0:515
¬ 22433/lpd Waiting
tcp 0 0 1.2.3.1:139
¬ 1746/smbd
tcp 0 0 0.0.0.0:111
¬ 1230/portmap
tcp 0 0 0.0.0.0:6000
¬ 3551/X
tcp 0 0 1.2.3.1:8081
¬ 18735/junkbuster
tcp 0 0 1.2.3.1:3128
¬ 18822/(squid)
tcp 0 0 127.0.0.1:953
¬ 30734/named
tcp 0 0 ::ffff:1.2.3.1:993
¬ 6742/xinetd-ipv6
tcp 0 0 :::13
¬ 6742/xinetd-ipv6
tcp 0 0 ::ffff:1.2.3.1:143
  ¬ 6742/xinetd-ipv6
tcp 0 ¬ 30734/named tcp 0 ¬ 1410/sshd tcp 0 ¬ 13237/sshd udp 0 ¬ 1258/rpc.statd

udp             0
¬
udp             0
```

ϟ 1502/rpc.mountd udp 0 ϟ -udp 0 ϟ 1751/nmbd udp 0 ϟ 1751/nmbd udp 0

0 :::53 0 :::22 0 :::6010 0 0.0.0.0:32768 0 0.0.0.0:2049 0 0.0.0.0:32770 0 0.0.0.0:32771
0 1.2.3.1:137
0 0.0.0.0:137
0 1.2.3.1:138

Foreign Address	State
0.0.0.0:*	LISTEN
0.0.0.0:*	LISTEN
0.0.0.0:*	LISTEN
0.0.0.0:*	LISTEN
0.0.0.0:*	LISTEN
0.0.0.0:*	LISTEN
0.0.0.0:*	LISTEN
0.0.0.0:*	LISTEN
0.0.0.0:*	LISTEN
:::*	LISTEN
:::*	LISTEN
:::*	LISTEN
:::*	LISTEN
:::*	LISTEN
:::*	LISTEN
0.0.0.0:*	
0.0.0.0:*	
0.0.0.0:*	
0.0.0.0:*	
0.0.0.0:*	
0.0.0.0:*	
0.0.0.0:*	

¬ 1751/nmbd

udp 0	0 0.0.0.0:138	0.0.0.0:*

¬ 1751/nmbd

udp 0	0 0.0.0.0:33044	0.0.0.0:*

¬ 30734/named

udp 0	0 1.2.3.1:53	0.0.0.0:*

¬ 30734/named

udp 0	0 127.0.0.1:53	0.0.0.0:*

¬ 30734/named

udp 0 0 0.0.0.0:67 0.0.0.0:*
¬ 1530/dhcpd

udp 0 0 0.0.0.0:67 0.0.0.0:*
¬ 1530/dhcpd

udp 0 0 0.0.0.0:32858 0.0.0.0:*
¬ 18822/(squid)

udp 0 0 0.0.0.0:4827 0.0.0.0:*
¬ 18822/(squid)

udp 0 0 0.0.0.0:111 0.0.0.0:*
¬ 1230/portmap

udp 0 0 :::53 :::*
¬ 30734/named

Příklady výpisů paketů příkazem tcdump

Uvádíme několik příkladů zachycených paketů, které se mohou hodit k ladění...

Zjištění směrovače

Oznámení směrovače

15:43:49.484751 fe80::212:34ff:fe12:3450 > ff02::1: icmp6: router ¬ advertisement(chlim=64, router_ltime=30, reachable_time=0, ¬ retrans_time=0)(prefix info: AR valid_ltime=30, preffered_ltime=20, ¬ prefix=2002:0102:0304:1::/64)(prefix info: LAR valid_ltime=2592000, ¬ preffered_ltime=604800, prefix=3ffe:ffff:0:1::/64)(src lladdr: ¬ 0:12:34:12:34:50) (len 88, hlim 255)

Směrovač s lokální adresou „fe80::212:34ff:fe12:3450“ zašle oznámení vícesměrovou adresou všem uzlům. Adresa „ff02::1“ obsahuje dva prefixy „2002:0102:0304:1::/64“ (doba životnosti 30 s) a „3ffe:ffff:0:1::/64“ (doba životnosti 2 592 000 s) včetně své fyzické adresy „0:12:34:12:34:50“.

Vyžádání směrovače

15:44:21.152646 fe80::212:34ff:fe12:3456 > ff02::2: icmp6: router solicitation ¬ (src lladdr: 0:12:34:12:34:56) (len 16, hlim 255)

Uzel s lokální adresou „fe80::212:34ff:fe12:3456“ a fyzickou adresou „0:12:34:12:34:56“ hledá na připojení směrovač, proto posílá tuto žádost všem směrovačům na připojení vícesměrovou adresou „ff02::2“.

Zjišování sousedů

Žádost o zjištění sousedů z důvodu detekce duplicitní adresy

Následující pakety jsou odeslány uzlem s fyzickou adresou „0:12:34:12:34:56“ v průběhu autokonfigurace, aby se ověřilo, zda potenciální adresu nepoužívá jiný uzel na spojení. Pakety jsou odeslány na vícesměrovou lokální adresu zkoumaných uzlů.

- Uzel, který chce zkonfigurovat svoji lokální adresu „fe80::212:34ff:fe12:3456“, si ověří možnou duplicitu.

  15:44:17.712338 :: > ff02::1:ff12:3456: icmp6: neighbor sol: who has ¬ fe80::212:34ff:fe12:3456(src lladdr: 0:12:34:12:34:56) (len 32, hlim 255)

- Uzel, který chce zkonfigurovat svoji globální adresu „2002:0102:0304:1:212:34ff:fe12:3456“ (poté co obdržel oznámení shora), si ověří možnou duplicitu.

  15:44:21.905596 :: > ff02::1:ff12:3456: icmp6: neighbor sol: who has ¬ 2002:0102:0304:1:212:34ff:fe12:3456(src lladdr: 0:12:34:12:34:56) (len 32, ¬ hlim 255)

- Uzel, který chce zkonfigurovat svoji globální adresu „3ffe:ffff:0:1:212:34ff:fe12:3456“ (poté co obdržel oznámení shora), si ověří možnou duplicitu.

  15:44:22.304028 :: > ff02::1:ff12:3456: icmp6: neighbor sol: who has ¬ 3ffe:ffff:0:1:212:34ff:fe12:3456(src lladdr: 0:12:34:12:34:56) (len 32, hlim ¬ 255)

Zjišování sousedů kvůli hledání počítače nebo brány

- Uzel chce poslat pakety na „3ffe:ffff:0:1::10“, avšak nemá fyzickou adresu, aby mohl posílat pakety, takže nejdříve pošle žádost.

  13:07:47.664538 2002:0102:0304:1:2e0:18ff:fe90:9205 > ff02::1:ff00:10: icmp6: ¬ neighbor sol: who has 3ffe:ffff:0:1::10(src lladdr: 0:e0:18:90:92:5) (len 32, ¬ hlim 255)

- Uzel teď hledá „fe80::10“.

  13:11:20.870070 fe80::2e0:18ff:fe90:9205 > ff02::1:ff00:10: icmp6: neighbor ¬ sol: who has fe80::10(src lladdr: 0:e0:18:90:92:5) (len

32, hlim 255)

Podpora stálé konfigurace IPv6 v linuxových distribucích

Některé linuxové distribuce už obsahují podporu stálé konfigurace IPv6 pomocí stávající nebo nové konfigurace a skriptů a pomocí úprav ve skriptech IPv4.

Red Hat Linux a „klony“

Od okamžiku, kdy jsem začal psát tento návod, bylo mým úmyslem dát k dispozici stálou konfi-guraci IPv6, která by pokrývala většinu požadovaných případů, jako je samostatný počítač, samo-statný směrovač, počítač připojený ke dvěma sítím, směrovač s druhou sítí, normální tunely, tune-ly 6to4 atd. Nyní existuje množina konfigurací a skriptů, které tyto požadavky splňují (nikdy jsem v této souvislosti neslyšel o vážných problémech, ale nevím, kolik uživatelů je využívá). Vzhledem k tomu, že tyto konfigurace a skripty jsou průběžně doplňovány, mají svoji domovskou stránku: *http://www.deepspace6.net/projects/initscripts-ipv6.html.* Vzhledem k tomu, že jsem svoje zkuše-nosti začal sbírat v klonu systému Red Hat Linux 5.0, moje vývojové systémy IPv6 jsou převážně založeny na systému Red Hat Linux a je vcelku logické, že skripty byly vyvinuty pro tuto distri-buci (tzv. *historické vydání*). Také bylo velmi jednoduché některé konfigurační soubory rozšířit, vytvořit nové anebo upravit nastavení IPv4 na IPv6.

Šťastnou náhodou se některé moje skripty dostaly do distribuce Red Hat Linux 7.1, což je zásluha Pekky Savoly.
Mandrake od verze 8.0 také obsahuje balík skriptů s podporou IPv6, avšak jejich využívání brání drobná chyba (ifconfig nemá inet6 před add).

Test, zda síové konfigurační skripty podporují IPv6

Pomocí mé množiny si můžete otestovat, jestli vaše linuxová distribuce obsahuje podporu trvalé konfigurace IPv6. V distribuci by měla být tato knihovna skriptů:

/etc/sysconfig/network-scripts/network-functions-ipv6

Automatický test:

test -f /etc/sysconfig/network-scripts/network-functions-ipv6 && echo ¬”Hlavní knihovna skriptů IPv6 existuje”

Pokud některé funkce chybějí, je důležité znát verzi knihovny. Můžete ji zjistit pomocí následujícího příkazu (anebo snadněji, když se podíváte na začátek souboru):

source /etc/sysconfig/network-scripts/network-functions-ipv6 && ¬ getversion_ipv6_functions 20011124

V uvedeném příkladu je použita verze 20011124. Porovnejte ji s nejnovější verzí uvedenou na *http://www.deepspace6.net/projects/initscripts-ipv6.html*, abyste zjistili, co se změnilo. Také zde naleznete údaj o datu změny.

Krátký tip, jak aktivovat IPv6 na stávajícím RHL 7.1, 7.2, 7.3, ...

- Ověřte si, zda má systém zavedené moduly IPv6

  # modprobe -c | grep net-pf-10 alias net-pf-10 off

- Je-li výsledkem „off“, aktivujte podporu sítí IPv6 tak, že do souboru /etc/sysconfig/net-work přidáte následující nový řádek:

  NETWORKING_IPV6=yes

- Podporu sítí znovu zaveďte nebo restartujte pomocí:

  # service network restart

- Nyní by měly být moduly IPv6 zavedeny:

modprobe -c | grep ipv6 alias net-pf-10 ipv6

Je-li váš systém na spojení, které zajišťuje ohlašování směrovačů, autokonfigurace se provede auto-maticky. Další informace o tom, které nastavení je podporováno, naleznete v souboru /usr/share/doc/initscripts-$verze/sysconfig.txt.

SuSE Linux

Ve verzích novějších než 7.x je skutečně jen základní podpora, podrobnosti viz /etc/rc.config. Vzhledem k velkým rozdílům v konfiguracích a ve struktuře skriptů je velmi obtížné (ne-li nemož-né) používat nástroje Red Hat Linuxu a klonů v této distribuci. Ve verzi 8.x byl dokonce zcela změněn způsob konfigurace.

SuSE Linux 7.3

- *http://www.feyrer.de/IPv6/SuSE73-IPv6+6to4-setup.html*

SuSE Linux 8.0

Konfigurace adres IPv6

Do souboru /etc/sysconfig/network/ifcfg-<jméno-rozhraní> nastavte hodnotu IP6ADDR="<adresa-ipv6>/<prefix>"

Doplňující informace

Viz soubor /usr/share/doc/packages/sysconfig/README.

SuSE Linux 8.1

Konfigurace adres IPv6

Do souboru /etc/sysconfig/network/ifcfg-<jméno-rozhraní> nastavte hodnotu IPADDR="< adresa-ipv6>/<prefix>"

Doplňující informace

Viz soubor /usr/share/doc/packages/sysconfig/Network.

Debian Linux

Následující informace poskytl Stephane Bortzmeyer <*bortzmeyer@nic.fr*>.

1. Přesvědčte se, zda je zaveden modul pro IPv6. Buď musel být přeložen společně s jádrem nebo musel být zaveden jako samostatný modul anebo přinejmenším bylo zavedení vyře-šeno jedním ze tří následujících způsobů: Modul byl přidán do /etc/modules, bylo pro-vedeno „předzavedení" (trik, který bude vysvětlen později) anebo byl zaveden pomocí kmod (uvádíme bez dalších podrobností).
2. Zkonfigurujte rozhraní. Předpokládejme eth0 a adresu (3ffe:ffff: 1234:5::1:1). Soubor /etc/network/interfaces upravte takto: iface eth0 inet6 static pre-up modprobe ipv6 address 3ffe:ffff:1234:5::1:1 # To suppress completely autoconfiguration: # up echo 0 > /proc/sys/net/ipv6/conf/all/autoconf netmask 64 # The router is autoconfigured and has no fixed address. # It is magically # found. (/proc/sys/net/ipv6/conf/all/accept_ra). Otherwise: #gateway 3ffe:ffff:1234:5::1

Nyní buď znovu zaveďte systém, nebo zadejte příkaz:

```
# ifup --force eth0
```

a získáte svoji statickou adresu.

Další informace

http://people.debian.org/~csmall/ipv6/, napsal Craig Small.
Jean-Marc V. Liotier: *http://www.ruwenzori.net/ipv6/Jims_LAN_IPv6_global_ connectivity_howto.html* (oznámeno 24. 12. 2002 v IPv6 konferenci účastníkem s adresou <*users@ipv6.org*>).

Autokonfigurace a mobilita

Bezstavová autokonfigurace

Poté co je rozhraní aktivované pro IPv6 připojeno, je podporováno a viditelné na přiřazené lokál
ní adrese.
Příklad:

```
# ip -6 addr show dev eth0 scope link
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qlen1000
    inet6 fe80::211:d8ff:fe6b:f0f5/64 scope link
       valid_lft forever preferred_lft forever
```

Stavová autokonfigurace pomocí démona pro oznamování směrovače (radvd)

Viz Autokonfigurace démona radvd v dalším textu.

Protokol DHCPv6

Po dlouhých diskusích byl konečně dokončen protokol RFC 3315 (Dynamic Host Configuration Protocol for IPv6, DHCPv6). V

době aktualizace této části (10/2005) byly k dispozici dvě implementace:

Dibbler (*http://klub.com.pl/dhcpv6/*) od Tomasze Mrugalského <*thomson@klub.com* dot pl>.
DHCPv6 on Sourceforge (*http://dhcpv6.sourceforge.net/*).

Mobilita

Momentálně jsou podrobnosti v *http://www.mipl.mediapoli.com/*. Další informace lze nalézt zde (prosím, ohlaste nefunkční odkazy):

http://www.ietf.org/internet-drafts/,
http://www.cs-ipv6.lancs.ac.uk/ipv6/MobileIP/,
http://gsyc.escet.urjc.es/Mobiquo/Mind/documentacion/MontajeMaquetaIPv6_en/Montaje-MaquetaIPv6_en.html,
http://www.piuha.net/~jarkko/publications/mipv6/MIPv6-Issues.html.

Firewally

Firewally IPv6 jsou důležité, zejména když se používá IPv6 na interních sítích s globálními adre-sami IPv6. Protože na rozdíl od IPv4, kde v běžných interních počítačích jsou globální adresy auto-maticky chráněny pomocí privátních adres IPv4, např. RFC 1918 nebo APIPA (Automatic Private IP Addressing, *http://www.google.com/search?q=apipa+microsoft*), v případě IPv6 se globální adre-sy obvykle používají a někdo s konektivitou IPv6 může obsáhnout všechny interní uzly s podpo-rou IPv6.

Firewally s filtry netfilter6

Vlastní firewally IPv6 jsou podporovány v jádru verze 2.4+. Ve starších jádrech 2.2- můžete používat pouze filtry IPv6-in-IPv4 podle protokolu 41.
Upozornění: Neexistuje záruka, že popsaná pravidla nebo příklady mohou skutečně ochránitsystém.
Podrobnosti o auditu pravidel po instalaci viz kapitola „Bezpečnostní audit IPv6“.Také poznamenejme, že projekt USAGI je v současné době před dokončením spojení pro IPv6!Díky tomu bude množina pravidel do budoucnosti jednodušší a bezpečnější.

Další informace

http://www.netfilter.org/,
https://lists.netfilter.org/mailman/listinfo/netfilter,
https://lists.netfilter.org/mailman/listinfo/netfilter-devel,
http://www.bieringer.de/linux/IPv6/status/IPv6+Linux-status-kernel.html#netfilter6.

Příprava Získání zdrojů

Obstarejte si nejnovější zdrojový kód jádra: *http://www.kernel.org/*. Obstarejte si nejnovější

balík iptables:

Zdrojový archív v taru (záplaty jádra): *http://www.netfilter.org/*.
Zdrojový balíček RPM pro vytvoření binárních souborů (systém RedHat): ftp://ftp.
redhat.com/redhat/linux/rawhide/SRPMS/SRPMS/ nebo možná také na *http://www.-netcore.fi/pekkas/linux/ipv6/*.

Rozbalení zdrojů

Přejděte do zdrojového adresáře:

```
# cd /path/to/src
```

Rozbalte a přejmenujte zdroje jádra:

```
# tar zxf kernel-version.tar.gz # mv linux linux-version-iptables-version+IPv6
```

Rozbalte zdroje iptables:

```
# tar zxf iptables-version.tar.gz
```

Aplikujte nejnovější záplaty iptables/IPv6 na zdroj jádra

Přejděte do zdrojového adresáře:

```
# cd iptables-version
```

Aplikujte stávající záplaty:

```
# make pending-patches KERNEL_DIR=/path/to/src/linux-version-iptables-version/
```

Aplikujte dodatečné záplaty IPv6 (prozatím nejsou součástí vanilla jádra):

```
# make patch-o-matic KERNEL_DIR=/path/to/src/linux-version-iptables-version/
```

Potvrďte následující volby (iptables-1.2.2):

ah-esp.patch,
masq-dynaddr.patch (pouze pro systémy s dynamickou IP adresou přiřazenou spojení WAN, např. PPP nebo PPPoE),
ipv6-agr.patch.ipv6,
ipv6-ports.patch.ipv6,
LOG.patch.ipv6,
REJECT.patch.ipv6.

Ověřte rozšíření IPv6:

```
# make print-extensions Extensions found: IPv6:owner IPv6:limit IPv6:mac IPv6:multiport
```

Konfigurace, vytvoření a instalace nového jádra

Přejděte do zdrojového adresáře:

```
# cd /path/to/src/linux-version-iptables-version/
```

Upravte Makefile:

```
-EXTRAVERSION =
+ EXTRAVERSION = -iptables-version+IPv6-try
```

Spusťte konfiguraci, aktivujte IPv6:

Code maturity level options Prompt for development and/or incomplete code/drivers:yes
Networking options
Network packet filtering: yes
The IPv6 protocol: module

IPv6: Netfilter Configuration IP6 tables support: module All new options like following:
limit match support: module MAC address match support: module Multiple port
match support: module Owner match support: module netfilter MARK match
support: module Aggregated address check: module Packet filtering: module
REJECT target support: module
LOG target support: module Packet mangling: module MARK target support: module

Zkonfigurujte zbytek systému.

Kompilace a instalace: viz část o jádru v tomto návodu a také v jiných návodech

Vytvoření a instalace binárního tvaru iptables

Přesvědčte se, že je dostupná i horní část stromu zdrojového kódu jádra na /usr/src/linux/. Přejmenujte starší adresář:

```
# mv /usr/src/linux /usr/src/linux.old
```

Vytvořte nový odkaz na software:

```
# ln -s /path/to/src/linux-version-iptables-version /usr/src/linux
```

Vytvořte nový SRPMS:

```
# rpm --rebuild /path/to/SRPMS/iptables-version-release.src.rpm
```

Nainstalujte nové balíky iptables (iptables + iptables-ipv6):

- V systémech RH 7.1 je obvykle už nainstalována starší verze, použijte proto parametr pro „freshen“ (aktualizaci).

  ```
  # rpm -Fhv /path/to/RPMS/cpu/iptables*-version-release.cpu.rpm
  ```

- Není-li už nainstalován, zadejte „install“:

  ```
  # rpm -ihv /path/to/RPMS/cpu/iptables*-version-release.cpu.rpm
  ```

- V systémech RH 6.2 obvykle není nainstalováno jádro 2.4.x, proto požadavky nevyhovují. Nainstalujte je pomocí --nodeps.

```
# rpm -ihv --nodeps /path/to/RPMS/cpu/iptables*-version-release.cpu.rpm
```

Pravděpodobně je nutné vytvořit odkaz na knihovny iptables, kde je iptables hledá:

```
# ln -s /lib/iptables/ /usr/lib/iptables
```

Použití Kontrola podpory

Zaveďte modul, je-li zkompilován:

```
# modprobe ip6_tables
```

Zkontrolujte podporu:

```
# [ ! -f /proc/net/ip6_tables_names ] && echo "Current kernel doesn't support ¬ 'ip6tables' firewalling (IPv6)!"
```

Naučte se používat ip6tables

Vypište si všechny položky síových filtrů IPv6

- Krátce:

  ```
  # ip6tables -L
  ```

- Podrobně:

  ```
  # ip6tables -n -v --line-numbers -L
  ```

Výpis určitého filtru

```
# ip6tables -n -v --line-numbers -L INPUT
```

Do vstupního filtru vložte logovací pravidla s volbami

```
# ip6tables --table filter --append INPUT -j LOG --log-prefix "INPUT:" ¬ --log-level 7
```

Do vstupního filtru vložte pravidla pro odmítání

```
# ip6tables --table filter --append INPUT -j DROP
```

Zrušte pravidlo dle čísla

```
# ip6tables --table filter --delete INPUT 1
```

Povolte ICMPv6

Se starším jádrem (jádro bez záplat 2.4.5 a iptables-1.2.2) nelze specifikovat typ.

- Akceptujte příchozí ICMPv6 prostřednictvím tunelů:

  ```
  # ip6tables -A INPUT -i sit+ -p icmpv6 -j ACCEPT
  ```

- Povolte odchozí ICMPv6 prostřednictvím tunelů:

  ```
  # ip6tables -A OUTPUT -o sit+ -p icmpv6 -j ACCEPT
  ```

Novější jádra umožňují specifikaci typů ICMPv6:

```
# ip6tables -A INPUT -p icmpv6 --icmpv6-type echo-request -j ACCEPT
```

Omezování rychlosti

Může se stát (autor to už viděl dvakrát), že dojde k útoku ICMPv6. Z toho důvodu byste měli pou-žívat dostupné omezování rychlosti alespoň na úrovni množiny pravidel ICMPv6. Navíc by měla být omezena i pravidla pro přihlašování, abyste zabránili útokům DoS na logy systému a ukládá-ní oblastí s logovacím souborem. Příklad omezení rychlosti ICMPv6 vypadá takto:

```
# ip6tables -A INPUT --protocol icmpv6 --icmpv6-type echo-request ¬ -j ACCEPT --match limit --limit 30/minute
```

Povolení příchozího SSH

Množina pravidel v uvedeném příkladu umožňuje příchozí spojení SSH z určité adresy IPv6.

- Povolení příchozího SSH z 3ffe:ffff:100::1/128:

  ```
  # ip6tables -A INPUT -i sit+ -p tcp -s 3ffe:ffff:100::1/128 --sport 512:65535 ¬ --dport 22 -j ACCEPT
  ```

- Povolení odpovědních paketů (v současné době není sledování spojení IPv6 v hlavní části netfilter6 implementováno):

  ```
  # ip6tables -A OUTPUT -o sit+ -p tcp -d 3ffe:ffff:100::1/128 --dport 512:65535 ¬ --sport 22 ! --syn -j ACCEPT
  ```

Aktivace tunelu IPv6-in-IPv4

Abyste mohli přijímat pakety z tunelu IPv6-in-IPv4, musíte do nastavení firewallu vložit pravidla, která se vztahují k těmto paketům, například:

Přijímej příchozí IPv6-in-IPv4 na rozhraní ppp0:
Povol odchozí IPv6-in-IPv4 na rozhraní ppp0:

```
# iptables -A INPUT -i ppp0 -p ipv6 -j ACCEPT

# iptables -A OUTPUT -o ppp0 -p ipv6 -j ACCEPT
```

Máte-li pouze statický tunel, můžete také specifikovat adresu IPv4, např.:

Přijímej příchozí IPv6-in-IPv4 na rozhraní ppp0 z koncového bodu tunelu 1.2.3.4:
Povol odchozí IPv6-in-IPv4 na rozhraní ppp0 na koncovém bodu tunelu 1.2.3.4:

```
# iptables -A INPUT -i ppp0 -p ipv6 -s 1.2.3.4 -j ACCEPT

# iptables -A OUTPUT -o ppp0 -p ipv6 -d 1.2.3.4 -j ACCEPT
```

Ochrana proti příchozím požadavkům na spojení TCP

VELMI DOPORUČUJEME! Z důvodu bezpečnosti byste skutečně měli vložit pravidlo, které bloku-je příchozí požadavky na spojení TCP. Jsou-li ostatní jména rozhraní použita, zadejte volbu -i.

- Blokování příchozích požadavků na spojení TCP na tomto počítači:

```
# ip6tables -I INPUT -i sit+ -p tcp --syn -j DROP
```

- Blokování příchozích požadavků na spojení TCP na počítačích za tímto směrovačem

```
# ip6tables -I FORWARD -i sit+ -p tcp --syn -j DROP
```

Pravidla musí pravděpodobně být umístěna za ostatními, avšak to je práce, kterou byste si měli dobře promyslet. Nejlepší je vytvořit skript a pravidla provádět uvedeným způsobem.

Ochrana proti příchozím požadavkům na spojení UDP

TAKÉ DOPORUČUJEME! Jak už jsem se zmínil v popisu firewallu, je možné řídit porty na odcho-zích sezeních UDP/TCP. Pokud tedy všechny lokální systémy IPv6 používají lokální porty napří-klad v rozmezí od 32 768 do 60 999, jste schopni filtrovat také spojení UDP (dokud funguje sle-dování spojení), například:

- Blokování příchozích paketů UDP, které nemohou být odpověďmi na odchozí požadavky tohoto počítače:

```
# ip6tables -I INPUT -i sit+ -p udp ! --dport 32768:60999 -j DROP
```

- Blokování příchozích paketů UDP, které nemohou být odpověďmi na přeposlané požadavky počítačů za tímto směrovačem:

```
# ip6tables -I FORWARD -i sit+ -p udp ! --dport 32768:60999 -j DROP
```

Demonstrační příklad

V následujícím příkladu naleznete sofistikovanější nastavení. Šťastné je vytváření množiny pravidel netfilter6...

```
# ip6tables -n -v -L
Chain INPUT (policy DROP 0 packets, 0 bytes)

  pkts bytes target prot opt in out source destination 0 0 extIN all sit+ * ::/0 ::/0 4 384 intIN all eth0 * ::/0 ::/0 0 0 ACCEPT all * * ::1/128 ::1/128
       0 0 ACCEPT all lo * ::/0 ::/0 0 0 LOG all * * ::/0 ::/0
¬ LOG flags 0 level 7 prefix `INPUT-default:' 0 0 DROP all * * ::/0 ::/0

Chain FORWARD (policy DROP 0 packets, 0 bytes) pkts bytes target prot opt in out source destination

       0 0 int2ext all eth0 sit+ ::/0 ::/0 0 0 ext2int all sit+ eth0 ::/0 ::/0 0 0 LOG all * * ::/0 ::/0
¬ LOG flags 0 level 7 prefix `FORWARD-default:' 0 0 DROP all * * ::/0 ::/0

Chain OUTPUT (policy DROP 0 packets, 0 bytes) pkts bytes target prot opt in out source destination 0 0 extOUT all * sit+ ::/0 ::/0 4 384 intOUT
  all * eth0 ::/0 ::/0 0 0 ACCEPT all * * ::1/128 ::1/128 0 0 ACCEPT all * lo ::/0 ::/0 0 0 LOG all * * ::/0 ::/0
¬ LOG flags 0 level 7 prefix `OUTPUT-default:' 0 0 DROP all * * ::/0 ::/0

Chain ext2int (1 references) pkts bytes target prot opt in out source destination
¬ 0 0 ACCEPT icmpv6 * * ::/0 ::/0 0 0 ACCEPT tcp * * ::/0 ::/0
¬ tcp spts:1:65535 dpts:1024:65535 flags:!0x16/0x02 0 0 LOG all * * ::/0 ::/0
¬ LOG flags 0 level 7 prefix `ext2int-default:' 0 0 DROP tcp * * ::/0 ::/0 0 0 DROP udp * * ::/0 ::/0 0 0 DROP all * * ::/0 ::/0

Chain extIN (1 references) pkts bytes target prot opt in out source destination
```

```
          0 0 ACCEPT tcp * * 3ffe:400:100::1/128 ::/0 ¬ tcp spts:512:65535 dpt:22 0 0 ACCEPT tcp * *
    3ffe:400:100::2/128 ::/0
¬ tcp spts:512:65535 dpt:22 0 0 ACCEPT icmpv6 * * ::/0 ::/0 0 0 ACCEPT tcp * * ::/0 ::/0
    ¬ tcp spts:1:65535 dpts:1024:65535 flags:!0x16/0x02 0 0 ACCEPT udp * * ::/0 ::/0 ¬ udp spts:1:65535 dpts:1024:65535
              0 0 LOG all * * ::/0 ::/0 ¬ limit: avg 5/min burst 5 LOG flags 0 level 7 prefix `extIN-default:' 0 0 DROP all * * ::/0 ::/0

   Chain extOUT (1 references) pkts bytes target prot opt in out source destination ¬ 0 0
     ACCEPT tcp * * ::/0 ¬ 3ffe:ffff:100::1/128tcp spt:22 dpts:512:65535 flags:!0x16/0x02 0 0
     ACCEPT tcp * * ::/0
¬ 3ffe:ffff:100::2/128tcp spt:22 dpts:512:65535 flags:!0x16/0x02 0 0 ACCEPT icmpv6 * * ::/0 ::/0 0 0 ACCEPT tcp * * ::/0 ::/0
    ¬ tcp spts:1024:65535 dpts:1:65535 0 0 ACCEPT udp * * ::/0 ::/0 ¬ udp spts:1024:65535 dpts:1:65535 0 0
          LOG all * * ::/0 ::/0 ¬ LOG flags 0 level 7 prefix `extOUT-default:' 0 0 DROP all * * ::/0 ::/0

Chain int2ext (1 references)
  pkts bytes target prot opt in out source destination ¬
       0 0 ACCEPT icmpv6 * * ::/0 ::/0
       0 0 ACCEPT tcp * * ::/0 ::/0 ¬ tcp spts:1024:65535 dpts:1:65535
       0 0 LOG all * * ::/0 ::/0 ¬ LOG flags 0 level 7 prefix `int2ext:'
       0 0 DROP all * * ::/0 ::/0
       0 0 LOG all * * ::/0 ::/0 ¬ LOG flags 0 level 7 prefix `int2ext-default:'
       0 0 DROP tcp * * ::/0 ::/0
       0 0 DROP udp * * ::/0 ::/0
       0 0 DROP all * * ::/0 ::/0

   Chain intIN (1 references) pkts bytes target prot opt in out source
     destination ¬ 0 0 ACCEPT all * * ::/0 ¬ fe80::/ffc0:: 4 384 ACCEPT all *
     * ::/0 ff02::/16

   Chain intOUT (1 references) pkts bytes target prot opt in out source
     destination ¬ 0 0 ACCEPT all * * ::/0
¬ fe80::/ffc0::
       4 384 ACCEPT all * * ::/0 ff02::/16
       0 0 LOG all * * ::/0 ::/0
             LOG flags 0 level 7 prefix `intOUT-default:'
       0 0 DROP all * * ::/0 ::/0
```

Bezpečnost

Bezpečnost uzlu

Velmi doporučujeme aplikovat všechny existující záplaty a zablokovat všechny nepotřebné služ-by. Také připojte služby pouze k adresám IPv4/IPv6 a nainstalujte lokální firewall.

Omezení přístupu

Mnohé služby používají pro řízení přístupu knihovnu tcp_wrapper. V dalším textu je popsáno použití tcp wrapperu, viz kapitola „Tcp_wrapper“.

Bezpečnostní audit IPv6

Momentálně nejsou k dispozici účinné nástroje, jimiž by bylo možné po síti zkontrolovat bezpeč-nostní opatření IPv6. Pokud vím, těmto účelům nevyhovuje ani Nessus (*http://www.nessus.org/*), ani žádný jiný komerční skener.

Zákonná úprava

UPOZORNĚNÍ: Nechcete-li porušit zákon, vždy dbejte na to, abyste skenovali pouze své vlastní systémy anebo jiné systémy jen na základě písemné objednávky. Než začnete se skenováním, cílovou adresu IPv6 si raději DVAKRÁT OVĚŘTE!

Bezpečnostní audit prováděný pomocí utility netcat s podporou IPv6

Pomocí nástroje netcat s podporou IPv6 (podrobnosti viz *http://www.bieringer.de/linux/IPv6/ status/IPv6+Linux-status-apps.html#security-auditing*) můžete spustit portscan tak, že zabalíte skript, aby prošel daný rozsah portů, zachytil bannery atd. Příklad použití:

```
# nc6 ::1 daytime 13 JUL 2002 11:22:22 CEST
```

Bezpečnostní audit prováděný pomocí nástroje nmap s podporou IPv6

NMap (*http://www.insecure.org/nmap/*), jeden z nejlepších skenerů portů na světě, podporuje IPv6 od verze 3.10ALPHA1. Příklad použití:

nmap -6 -sT ::1Starting nmap V. 3.10ALPHA3 (www.insecure.org/nmap/) Interesting ports on localhost6 (::1): (The 1600 ports scanned but not shown below are in state: closed)

Port	State	Service
22/tcp	open	ssh
53/tcp	open	domain
515/tcp	open	printer
2401/tcp	open	cvspserver

Nmap run completed -- 1 IP address (1 host up) scanned in 0.525 seconds

Bezpečnostní audit prováděný pomocí nástroje strobe s podporou IPv6

Strobe je (ve srovnání s NMap) jednodušší, existuje záplata pro podporu IPv6 (podrobnosti viz *http://www.bieringer.de/linux/IPv6/status/IPv6+Linux-status-apps.html#security-auditing*). Příklad použití:

./strobe ::1 strobe 1.05 (c) 1995-1999 Julian Assange <proff@iq.org>. ::1 2401 unassigned unknown ::1 22 ssh Secure Shell - RSA encrypted rsh ::1 515 printer spooler (lpd) ::1 6010 unassigned unknown ::1 53 domain Domain Name Server

Poznámka: Vývoj strobe už nepokračuje, uvedená verze není správná.

Výsledky auditu

Jestliže výsledky auditu nejsou ve shodě s vaší bezpečnostní politikou IPv6, zabezpečte zranitel-ná místa firewallem IPv6 s použitím netfilter6 (podrobnosti viz kapitola „Firewally“).Informace: Podrobnější informace o bezpečnosti IPv6 naleznete na adrese:

■ *http://www.ietf.org/internet-drafts/.*

Šifrování a autentizace

Na rozdíl od IPv4 je šifrování a autentizace povinnou součástí IPv6. Tyto funkce jsou obvykle implementovány pomocí IPsec (jež lze použít i v případě IPv4).

Režimy šifrování a autentizace

Existují dva režimy šifrování a autentizace spojení:

Transportní režim

Transportní režim je skutečný režim koncového spojení. Zašifrován je pouze obsah (obvykle ICMP, TCP nebo UDP) s hlavičkou, zatímco hlavička IP není zašifrovaná (avšak obvykle je součástí autentizace).

Transportní režim se šifrováním AES-128 a autentizací SHA1 snižuje MTU o 42 oktetů.

Tunelový režim

Tunelový režim je využíván buď pro koncové spojení nebo pro spojení mezi branami. V tomto případě jsou zašifrovány celé IP pakety a jsou opatřeny novými IP hlavičkami, čímž je vytvořen nový paket (tento mechanismus je znám pod názvem „zapouzdření“, angl. „encapsulation“).

Tento režim obvykle snižuje MTU o 40 oktetů oproti transportnímu režimu, tj. se šifrováním AES-128 a autentizací SHA1 je MTU o 82 oktetů nižší než normální MTU.

Podpora v jádru (ESP a AH) Podpora v jádru vanilla

Linuxu 2.4.x

V době psaní chyběla podpora IPv6 v jádru systému vanilla do verze 2.4.28. Kvůli exportním zákonům vztahujícím se na šifrování vznikl problém s volným šířením jádra. Toto je další případ, kdy do zdrojového kódu vanilla nebyl zahrnut projekt FreeS/WAN (*http://www.freeswan.org/*). Snad bude v budoucnosti toto jádro doplněno z jádra 2.6.x.

Podpora v jádru vanilla Linuxu 2.6.x

Současné verze (v době psaní 2.6.9 a vyšších) podporují vlastní IPsec pro IPv4 a IPv6. Pomoc při implementaci přišla i z projektu USAGI.

Automatická výměna klíčů (IKE)

V IPsec je požadována výměna klíčů. To se většinou děje automaticky pomocí tzv. démonů IKE. Také zpracovávají autentizaci účastníků buď pomocí sdílených klíčů (angl. „pre-shared secret“) nebo klíčů RSA (je možné je získat i z certifikátů X.509).

V Linuxu jsou v současnosti k dispozici dva různé démony IKE, které se podstatně liší jak v kon
figuraci, tak i způsobem použití.Já dávám přednost démonu *pluto* v implementaci *S/WAN, neboť je jednodušší a vyžaduje pouzejednu konfiguraci.

IKE démon racoon

IKE démon *racoon* byl převzat z projektu KAME a přenesen do Linuxu. V současných linuxových distribucích jsou součástí balíku ipsec-tools. Ke správnému nastavení IPsec jsou potřebné dva pro-gramy. Viz také *http://lartc.org/howto/lartc.ipsec.html*.

Manipulace s databází IPsec SA/SP pomocí nástroje setkey

Nástroj setkey je důležitý pro definici bezpečnostní politiky jádra. Soubor: /etc/racoon/setkey.sh

- Příklad šifrovaného koncového spojení v transportním režimu:

```
#!/sbin/setkey -f
flush;
spdflush;
spdadd 2001:db8:1:1::1 2001:db8:2:2::2 any -P out ipsec esp/transport//require;
spdadd 2001:db8:2:2::2 2001:db8:1:1::1 any -P in ipsec esp/transport//require;
```

- Příklad šifrovaného koncového spojení v tunelovém režimu:

```
#!/sbin/setkey -fflush;spdflush;spdadd 2001:db8:1:1::1 2001:db8:2:2::2 any -P out ipsec¬
esp/tunnel/2001:db8:1:1::1-2001:db8:2:2::2/require;spdadd 2001:db8:2:2::2 2001:db8:1:1::1 any -P in ipsec¬
esp/tunnel/2001:db8:2:2::2-2001:db8:1:1::1/require;
```

U ostatních účastníků musíte místo „in“ dát „out“.

Konfigurace IKE démona racoon

Pro správnou činnost vyžaduje racoon konfigurační soubor. Obsahuje příslušná nastavení bez-pečnostní politiky, která by měla být nastavena pomocí setkey už dřív.

Soubor: /etc/racoon/racoon.conf # Racoon IKE daemon configuration file. # See 'man racoon.conf' for a description of the format and entries. path include "/etc/racoon"; path pre_shared_key "/etc/racoon/psk.txt"; listen { isakmp 2001:db8:1:1::1; } remote 2001:db8:2:2::2 { exchange_mode main; lifetime time 24 hour; proposal { encryption_algorithm 3des; hash_algorithm md5; authentication_method pre_shared_key; dh_group 2; } } # gateway-to-gateway sainfo address 2001:db8:1:1::1 any address 2001:db8:2:2::2 any { lifetime time 1 hour; encryption_algorithm 3des; authentication_algorithm hmac_md5; compression_algorithm deflate; } sainfo address 2001:db8:2:2::2 any address 2001:db8:1:1::1 any {

```
    lifetime time 1 hour;
    encryption_algorithm 3des;
    authentication_algorithm hmac_md5;
    compression_algorithm deflate;
```

}

Nastavení sdíleného klíče: Soubor: /etc/racoon/psk.txt

file for pre-shared keys used for IKE authentication # format is: ‘identifier’ ‘key’ 2001:db8:2:2::2 verysecret

Spuštění IPsec s IKE démonem racoon

Nakonec je třeba démona spustit. Při prvním spuštění použijte ladicí program a démona spusťte na pozadí. Následující příklad znázorňuje úspěšnou komunikaci ve fázi 1 IKE (ISAKMP-SA) a 2 (IPsec-SA):

racoon -F -v -f /etc/racoon/racoon.confForeground mode. 2005-01-01 20:30:15: INFO: @(#)ipsec-tools 0.3.3 (http://ipsec-tools.sourceforge.net)2005-01-01 20:30:15: INFO: @(#)This product linked¬ OpenSSL 0.9.7a Feb 19 2003 (http://www.openssl.org/)2005-01-01 20:30:15: INFO: 2001:db8:1:1::1[500] used as isakmp port (fd=7)2005-01-01 20:31:06: INFO: IPsec-SA request for 2001:db8:2:2::2¬ queued due to no

phase1 found.2005-01-01 20:31:06: INFO: initiate new phase 1 negotiation:¬ 2001:db8:1:1::1[500]<=>2001:db8:2:2::2[500]2005-01-01 20:31:06: INFO: begin Identity Protection mode.2005-01-01 20:31:09: INFO: ISAKMP-SA established¬ 2001:db8:1:1::1[500]-2001:db8:2:2::2[500]spi:da3d3693289c9698:ac039a402b2db4012005-01-01 20:31:09: INFO: initiate new phase 2 negotiation:¬ 2001:6f8:900:94::2[0]<=>2001:db8:2:2::2[0]2005-01-01 20:31:10: INFO: IPsec-SA established:¬ ESP/Tunnel 2001:db8:2:2::2->2001:db8:1:1::1 spi=253935531(0xf22bfab) 2005-01-01 20:31:10: INFO: IPsec-SA established:¬ ESP/Tunnel 2001:db8:1:1::1->2001:db8:2:2::2 spi=175002564(0xa6e53c4)

Každý směr dostal vlastní IPsec-SA (jak je definováno ve standardu IPsec). Když provedete tcdump na příslušné rozhraní, jako výsledek IPv6 ping uvidíte:

20:35:55.305707 2001:db8:1:1::1 > 2001:db8:2:2::2:
ESP(spi=0x0a6e53c4,seq=0x3)
20:35:55.537522 2001:db8:2:2::2 > 2001:db8:1:1::1:
ESP(spi=0x0f22bfab,seq=0x3)

Zadaná SPI jsou dle předpokladu použita zde. Stávající aktivní parametry si

vypíšeme pomocí setkey:

```
# setkey -D 2001:db8:1:1::1 2001:db8:2:2::2 esp
mode=tunnel spi=175002564(0x0a6e53c4)
reqid=0(0x00000000)
                    E: 3des-cbc bd26bc45 aea0d249 ef9c6b89 7056080f 5d9fa49c 924e2edd
A: hmac-md5 60c2c505 517dd8b7 c9609128 a5efc2db seq=0x00000000 replay=4 flags=0x00000000 state=mature
      created: Jan    1 20:31:10 2005        current: Jan 1 20:40:47 2005
       diff: 577(s)   hard: 3600(s)          soft: 2880(s)
     last: Jan     1 20:35:05 2005           hard: 0(s)          soft: 0(s)
```

current: 540(bytes) hard: 0(bytes) soft: 0(bytes) allocated: 3 hard: 0 soft: 0 sadb_seq=1 pid=22358 refcnt=0
2001:db8:2:2::2 2001:db8:1:1::1 esp mode=tunnel spi=253935531(0x0f22bfab) reqid=0(0x00000000)
E: 3des-cbc c1ddba65 83debd62 3f6683c1 20e747ac 933d203f 4777a7ce
A: hmac-md5 3f957db9 9adddc8c 44e5739d 3f53ca0e seq=0x00000000 replay=4 flags=0x00000000 state=mature created: Jan 1 20:31:10 2005 current: Jan 1 20:40:47 2005 diff: 577(s) hard: 3600(s) soft: 2880(s) last: Jan 1 20:35:05 2005 hard: 0(s) soft: 0(s) current: 312(bytes) hard: 0(bytes) soft: 0(bytes) allocated: 3 hard: 0 soft: 0 sadb_seq=0 pid=22358 refcnt=0

IKE démon pluto

IKE démon pluto je součástí distribuce projektu *S/WAN, který se dříve jmenoval FreeS/WAN. Další vývoj tohoto projektu byl bohužel zastaven v roce 2004. Kvůli pomalému tempu vývoje v minulosti se od hlavního projektu oddělily dva vedlejší produkty: StrongSwan (*http://www.strongswan.org/*) a Openswan (*http://www.openswan.org/*). V současnosti je k dispo-zici přinejmenším instalační balík Openswan (je součástí Fedora Core 3).

Hlavním rozdílem oproti racoon je nutnost pouze jednoho konfiguračního souboru. Automatické nastavení po zavedení systému se provede pomocí inicializačního skriptu.

Konfigurace IKE démona pluto

Konfigurace je velice podobná konfiguraci v IPv4, pouze jedna volba je důležitá.

Soubor: /etc/ipsec.conf # /etc/ipsec.conf - Openswan IPsec configuration file # # Manual: ipsec.conf.5 version 2.0 # conforms to second version of ipsec.conf specification # basic configuration config setup # Debug-logging controls: "none" for (almost) none, "all" for lots. # klipsdebug=none # plutodebug="control parsing" #Disable Opportunistic Encryption include /etc/ipsec.d/examples/no_oe.conf conn ipv6-p1-p2 connaddrfamily=ipv6 # Important for IPv6! left=2001:db8:1:1::1 right=2001:db8:2:2::2 authby=secret esp=aes128-sha1 ike=aes128-sha-modp1024 type=transport

#type=tunnel compress=no #compress=yes auto=add #auto=start

Nezapomeňte zde také definovat sdílený klíč.Soubor: /etc/ipsec.secrets2001:db8:1:1::1 2001:db8:2:2::2 : PSK “verysecret”

Spuštění IPsec s IKE démonem pluto

Byla-li instalace Openswan úspěšná, pro spuštění IPsec by měl existovat inicializační skript. Na každém počítači pouze zadejte:

/etc/rc.d/init.d/ipsec start

Poté spusťte toto spojení na jednom počítači. Pokud zahlédnete řádek „IPsec SA established“, funguje vše správně.

ipsec auto --up ipv6-peer1-peer2 104 "ipv6-p1-p2" #1: STATE_MAIN_I1: initiate 106 "ipv6-p1-p2" #1: STATE_MAIN_I2: sent MI2, expecting MR2 108 "ipv6-p1-p2" #1: STATE_MAIN_I3: sent MI3, expertčin MR3 004 "ipv6-p1-p2" #1: STATE_MAIN_I4: ISAKMP SA established 112 "ipv6-p1-p2" #2: STATE_QUICK_I1: initiate 004 "ipv6-p1-p2" #2: STATE_QUICK_I2: sent QI2, ¬ IPsec SA established {ESP=>0xa98b7710 <0xa51e1f22}

Vzhledem k tomu, že *S/WAN a setkey/racoon používají v jádru Linuxu 2.6.x stejnou implemen-taci, můžeme zde použít setkey, abychom si ukázali stávající aktivní parametry:

```
    # setkey -D 2001:db8:1:1::1 2001:db8:2:2::2 esp
    mode=transport spi=2844489488(0xa98b7710)
    reqid=16385(0x00004001)
E: aes-cbc 082ee274 2744bae5 7451da37 1162b483
A: hmac-sha1 b7803753 757417da 477b1c1a 64070455 ab79082c seq=0x00000000 replay=64 flags=0x00000000 state=mature
        created: Jan     1 21:16:32 2005     current: Jan     1 21:22:20 2005
        diff: 348(s)     hard: 0(s)          soft: 0(s)
        last:                                hard: 0(s)          soft: 0(s)
```

current: 0(bytes) hard: 0(bytes) soft: 0(bytes) allocated: 0 hard: 0 soft: 0 sadb_seq=1 pid=23825 refcnt=0
2001:db8:2:2::2 2001:db8:1:1::1 esp mode=transport spi=2770214690(0xa51e1f22) reqid=16385(0x00004001)
E: aes-cbc 6f59cc30 8d856056 65e07b76 552cac18
A: hmac-sha1 c7c7d82b abfca8b1 5440021f e0c3b335 975b508b seq=0x00000000 replay=64 flags=0x00000000 state=mature created: Jan 1 21:16:31 2005 current: Jan 1 21:22:20 2005 diff: 349(s) hard: 0(s) soft: 0(s) last: hard: 0(s) soft: 0(s) current: 0(bytes) hard: 0(bytes) soft: 0(bytes) allocated: 0 hard: 0 soft: 0 sadb_seq=0 pid=23825 refcnt=0

Doplňující informace:

V linuxovém jádru 2.6.x můžete získat bezpečnostní politiku a stav IPsec také pomocí ip:

ip xfrm policy ... # ip xfrm state ...

Kvalita služby (QoS)

IPv6 podporuje QoS pomocí Flow Labels a Traffic Classes. K řízení můžete použít tc (součást balí-ku iproute).Doplňující informace:

- RFC 3697.

Démoni s podporou IPv6

Nabízíme některé tipy na démony s podporou IPv6.

BIND (Berkeley Internet Name Daemon, named)

IPv6 je podporováno od verze 9. Vždy používejte nejnovější možnou verzi, a to nejméně 9.1.3. Starší verze mohou obsahovat zranitelná místa zneužitelná po síti.

Naslouchání na adresách IPv6

Poznámka

Na rozdíl od IPv4 stávající verze neumožňuje svázat soket na serveru s určitou adresou IPv6, takže platné adresy jsou pouze *libovolná* nebo *žádná*. Vzhledem k tomu, že může jít o bezpečnostní problém, přečtěte si i část Access Control List (ACL) v dalším textu.

Aktivace démona BIND named pro naslouchání na adrese IPv6

Aktivace IPv6 pro naslouchání vyžaduje následující změnu voleb:

```
options { # sure other options here, too listen-on-v6 { any; };
};
```

Výsledek po restartu by měl být následující:

```
# netstat -lnptu |grep "named\W*$" tcp 0 0 :::53 :::* LISTEN 1234/named ¬ # incoming TCP requests udp 0 0 1.2.3.4:53 0.0.0.0:* 1234/named ¬
# incoming UDP requests to IPv4 1.2.3.4 udp 0 0 127.0.0.1:53 0.0.0.0:* 1234/named ¬ # incoming UDP requests to IPv4 localhost udp 0 0
0.0.0.0:32868 0.0.0.0:* 1234/named ¬ # dynamic chosen port for outgoing queries udp 0 0 :::53 :::* 1234/named ¬ # incoming UDP request to
any IPv6
```

A jednoduchý test vypadá takto:

```
# dig localhost @::1
```

Poté by se měl vypsat výsledek.

Blokování démona BIND named pro naslouchání na adrese IPv6:

Blokování IPv6 pro naslouchání vyžaduje následující změnu voleb

```
options { # sure other options here, too listen-on-v6 { none; };
};
```

Seznamy přístupových práv (Access Control Lists, ACL) s aktivací IPv6

ACL s aktivací IPv6 by se měly používat všude tam, kde je to možné. Například:

```
acl internal-net { 127.0.0.1; 1.2.3.0/24; 3ffe:ffff:100::/56; ::
        1/128; ::ffff:1.2.3.4/128;
};
acl ns-internal-net { 1.2.3.4; 1.2.3.5; 3ffe:ffff:100::4/128; 3ffe:ffff:100::5/128;
};
```

ACL se mohou používat například při dotazování klientů nebo přenosové zóny pomocných jmenných serverů. Tím zabráníte využívání jmenných serverů IPv6 zvenčí.

```
options { # sure other options here, too listen-on-v6 { none; }; allow-query { internal-net; }; allow-transfer { ns-internal-net; };
};
```

Je také možné nastavit volby allow-query a allow-transfer pro většinu definovaných samo-statných zón.

Posílání dotazů s jednoúčelovou adresou IPv6

Tato volba není povinná, avšak pravděpodobně je nutno ji uvést:

```
query-source-v6 address <ipv6address|*> port <port|*>;
```

Jednoúčelové adresy IPv6 definované pro zóny

Adresy IPv6 mohou být definovány pro zóny.

Přenos zdrojové adresy

Přenos zdrojové adresy se používá pro přenosy ze zón:

```
transfer-source-v6 <ipv6addr|*> [port port];
```

Oznámení zdrojové adresy

Oznámení zdrojové adresy se používá pro odchozí oznamovací zprávy:

```
notify-source-v6 <ipv6addr|*> [port port];
```

Příklady zónových souborů DNS IPv6

Některé informace naleznete i na *http://www.isi.edu/~bmanning/v6DNS.html.* Něco užitečného asi najdete i na *http://tools.fpsn.net/ipv6-inaddr/.*

Poskytování DNS dat IPv6

Pro nové typy IPv6 a kořenové zóny pro inverzní vyhledávání jsou definovány:

AAAA a inverzní IP6.INT: specifikované v RFC 1886, využitelné od verze BIND 4.9.6.
A6, DNAME (NYNÍ NESCHVÁLENÉ!) a inverzní IP6.ARPA: specifikované v RFC 2874, vyu-žitelné od verze BIND 9, viz také informace o současném stavu na *http://www.ietf.org/ internet-drafts/.*

Později bude pravděpodobně doplněno, prozatím si prohlédněte stávající RFC a také:

AAAA a inverzní IP6.INT: *http://www.isi.edu/~bmanning/v6DNS.html.*
A6, DNAME (NYNÍ NESCHVÁLENÉ!) a inverzní IP6.ARPA: nahlédněte do kapitol 4 a 6 dokumentu BIND 9 Administrator Reference Manual (ARM) distribuovaného s balíkem bind nebo na adrese:
http://www.nominum.com/content/documents/bind9arm.pdf.

Vzhledem k tomu, že IP6.INT není schválené (avšak používá se), DNS server, který bude podporovat informace IPv6, musí obsluhovat obě inverzní zóny.

Současná nejlepší praxe

Vzhledem k tomu, že s používáním nových formátů jsou určité problémy, současná nejlepší praxe je: Podpora přímého hledání: AAAA. Podpora inverzního hledání:
Inverzní „nibble“ (čtyřbitový) formát pro zónu ip6.int (PRO ZPĚTNOU KOMPATIBILITU).
Inverzní „nibble“ formát pro zónu ip6.arpa (DOPORUČENO).

Ověření spojení s aktivací IPv6

Jak ověřit, jestli BIND naslouchá na soketu IPv6 a poskytuje data, se dozvíte z následujících příkladů.

Spojení IPv6 odmítnuté ACL

Specifikací jednoúčelového serveru pro dotazy lze vytvořit spojení:

$ host -t aaaa www.6bone.net 3ffe:ffff:200:f101::1 Using domain server: Name: 3ffe:ffff:200:f101::1 Address: 3ffe:ffff:200:f101::1#53 Aliases:Host www.6bone.net. not found: 5(REFUSED)

Odpovídající položka v logu vypadá takto:

Jan 3 12:43:32 gate named[12347]: client ¬ 3ffe:ffff:200:f101:212:34ff:fe12:3456#32770:
query denied

Pokud v logu uvidíte takové položky, ověřte, zda dotazy od tohoto klienta jsou povolené, a také asi bude nutné zkontrolovat konfiguraci ACL.
Úspěšné spojení IPv6

Úspěšné spojení IPv6 vypadá takto:

$ host -t aaaa www.6bone.net 3ffe:ffff:200:f101::1 Using domain server: Name: 3ffe:ffff:200:f101::1 Address: 3ffe:ffff:200:f101::1#53 Aliases: www.6bone.net. is an alias for 6bone.net. 6bone.net. has AAAA address 3ffe:b00:c18:1::10

Internetový super démon (xinetd)

Xinetd (*http://www.xinetd.org/*) podporuje IPv6 od verze asi 1.8.9. Vždy používejte nejnovější možnou verzi, a to nejméně 2.3.3. Starší verze mohou obsahovat zranitelná místa zneužitelná po síti. Některé linuxové distribuce obsahují zvláštní balík pro xinetd s aktivovanou IPv6, některé jiné spouštějí xinetd s aktivovanou IPv6 tehdy, je-li nastavená proměnná NETWORKING_IPV6 na hod-notu yes, přičemž spuštění většinou provede /etc/sysconfig/network (což však platí pouze v distribucích Red Hat a odvozených). V novějších vydáních je podpora IPv4 i IPv6 zajištěna v jedi-ném binárním souboru.

Pokud aktivujete vestavěnou službu, např. daytime, modifikací konfiguračního souboru v /etc/xinetd.d/daytime, například takto: # diff -u /etc/xinetd.d/daytime.orig /etc/xinetd.d/daytime --- /etc/xinetd.d/daytime.orig Sun Dec 16 19:00:14 2001
+++ /etc/xinetd.d/daytime Sun Dec 16 19:00:22 2001
@@ -10,5 +10,5 @@ protocol = tcp user = root wait = no
-disable = yes
+ disable = no }
po restartu xinetd byste měli obdržet pozitivní výsledek, např.:

netstat -lnptu -A inet6 |grep "xinetd*" tcp 0 0 ::ffff:192.168.1.1:993 :::* LISTEN 12345/xinetd-ipv6 tcp 0 0 :::13 :::* LISTEN 12345/xinetd-ipv6 <- service ¬ daytime/tcp tcp 0 0 ::ffff:192.168.1.1:143 :::* LISTEN 12345/xinetd-ipv6

V uvedeném příkladu je také výpis xinetd, který naslouchá IMAP a IMAP-SSL s IPv4.

> Poznámka
>
> Ve starších verzích nebylo možné spustit xinetd/IPv4 na uzlu IPv6 a naopak xinetd/IPv6 nebylo možné spustit na uzlu IPv4. Chyba byla v novějších verzích odstraněna, přinejmenším od verze 2.3.11.

Webový server Apache2 (httpd2)

Webový server Apache podporuje IPv6 od verze 2.0.14. Záplaty pro starší verze řady 1.3.x nejsou aktualizované a neměly by se používat ve veřejném prostoru. Lze je nicméně nalézt na ftp://ftp.kame.net/pub/kame/misc/.

Naslouchání na adresách IPv6

> Poznámka
>
> Do verze 2.0.28 nefungují virtuální počítače na adresách IPv6 (pro verzi 2.0.28 existuje záplata). Vždy však nejdříve vyzkoušejte nejnovější verzi, neboť starší verze mají určité bezpečnostní problémy.

Virtuální počítač naslouchá pouze na adrese IPv6

```
Listen [3ffe:ffff:100::1]:80
<VirtualHost [3ffe:ffff:100::1]:80> ServerName ipv6only.yourdomain.yourtopleveldomain # ...sure more config lines
</VirtualHost>
```

Virtuální počítač naslouchá na adrese IPv6 i na adrese IPv4

```
Listen [3ffe:ffff:100::2]:80 Listen 1.2.3.4:80 <VirtualHost [3ffe:ffff:100::2]:80 1.2.3.4:80>
    ServerName ipv6andipv4.yourdomain.yourtopleveldomain # ...sure more config lines
    </VirtualHost>
```

Po restartu by měl být výsledek:

```
# netstat -lnptu |grep “httpd2\W*$” tcp 0 0 1.2.3.4:80 0.0.0.0:* LISTEN 12345/httpd2 tcp 0 0 3ffe:ffff:100::1:80 :::* LISTEN 12345/httpd2 tcp 0
0 3ffe:ffff:100::2:80 :::* LISTEN 12345/httpd2
```

K jednoduchému testu použijte příklad s příkazem telnet shora.

Doplňující poznámka

- Kvůli urychlení přenosu podporuje Apache2 metodu zvanou „sendfile“. Některé ovladače síťových karet také podporují kontrolní součty prováděné off-line. V některých případech to může vést k problémům se spojením a chybným kontrolním součtům TCP. V takovém případě zablokujte vlastnost „sendfile“ buď novou kompilací s volbou --without-sendfile nebo do konfiguračního souboru doplňte direktivu `EnableSendfile off`.

Démon pro oznamování směrovačů (radvd)

Démon pro oznamování směrovačů je v LAN velice užitečný, má-li být provedena autokonfigura-ce klienta. Démon samotný by měl běžet na implicitním směrovači brány IPv6 (tato brána však nemusí být současně implicitní bránou IPv4, takže věnujte pozornost tomu, kdo na vaší lokální síti posílá oznámení směrovače).

Můžete uvést určité informace a příznaky, které by mělo oznámení obsahovat. Obvykle se uvádí:

Prefix (povinný).
Životnost prefixu.
Četnost odesílání oznámení (volitelné).

Poté co je démon správně zkonfigurovaný, odešle oznámení přes uvedená rozhraní, klienti je snad obdrží a automaticky zkonfigurují adresy se zaslaným prefixem a s implicitní trasou.

Konfigurace radvd

Jednoduchá konfigurace

Konfigurační soubor radvd se obvykle jmenuje /etc/radvd.conf. Zde je jednoduchý příklad:

```
interface eth0
{ AdvSendAdvert on; MinRtrAdvInterval 3;
MaxRtrAdvInterval 10; prefix 3ffe:ffff:0100:f101::/64
{ AdvOnLink on; AdvAutonomous on; AdvRouterAddr on; };
};
```

Výsledek na straně klienta pak vypadá takto:

```
# ip -6 addr show eth0
3: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100 inet6 3ffe:ffff:100:f101:2e0:12ff:fe34:1234/64 scope
    global dynamic
        valid_lft 2591992sec preferred_lft 604792sec
    inet6 fe80::2e0:12ff:fe34:1234/10 scope link
```

Vzhledem k tomu, že nebyla definovaná životnost, dosadí se vysoká hodnota.

Speciální konfigurace 6to4

Verze od 0.6.2pl3 podporují automatické (re)generování prefixu v závislosti na adrese IPv4 na určeném rozhraní. Toho lze využít při distribuci oznámení v lokální síti po změně tunelu 6to4. Vět-šinou se využívá za dynamickým vytáčeným směrovačem. Kvůli

zaručeně kratší době životnosti takového prefixu (po každém vytočení je platný jiný prefix) je životnost zkonfigurovaná na minimální hodnotu:

```
interface eth0 { AdvSendAdvert on; MinRtrAdvInterval 3; MaxRtrAdvInterval 10; prefix 0:0:0:f101::/64 {
                        AdvOnLink off;
                        AdvAutonomous on;
                        AdvRouterAddr on;
                        Base6to4Interface ppp0;
                        AdvPreferredLifetime 20;
                        AdvValidLifetime 30;

        }; };
```

Tento výsledek na straně klienta je (předpokládejme, že ppp0 má lokální adresu IPv4 1.2.3.4):

```
# /sbin/ip -6 addr show eth0
3: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100 inet6 2002:0102:0304:f101:2e0:12ff:fe34:1234/64
    scope global dynamic
       valid_lft 22sec preferred_lft 12sec
    inet6 fe80::2e0:12ff:fe34:1234/10 scope link
```

Nepřijde-li příslušné oznámení, bude prefix vzhledem ke krátké životnosti záhy odstraněn.

Doplňující poznámka

Pokud v inicializačních skriptech nepoužíváte zvláštní podporu 6to4, musíte na interním rozhraní směrovače nastavit speciální trasu, jinak vzniknou problémy se zpáteční trasou, viz příklad:

```
# /sbin/ip -6 route add 2002:0102:0304:f101::/64 dev eth0 metric 1
```

Trasa se musí při každé změně prefixu také změnit, což je případ přiřazení adresy IPv4 vytáčenému rozhraní.

Ladění

Odeslaná a přijatá oznámení si můžete prohlížet pomocí programu radvdump. Použití je jedno-duché:

```
# radvdump
Router advertisement from fe80::280:c8ff:feb9:cef9 (hoplimit 255) AdvCurHopLimit: 64 AdvManagedFlag: off AdvOtherConfigFlag: off
        AdvHomeAgentFlag: off AdvReachableTime: 0 AdvRetransTimer: 0 Prefix 2002:0102:0304:f101::/64
                AdvValidLifetime: 30
                AdvPreferredLifetime: 20
                AdvOnLink: off
                AdvAutonomous: on
                AdvRouterAddr: on

        Prefix 3ffe:ffff:100:f101::/64
                AdvValidLifetime: 2592000
                AdvPreferredLifetime: 604800
                AdvOnLink: on
                AdvAutonomous: on
                AdvRouterAddr: on

        AdvSourceLLAddress: 00 80 12 34 56 78
```

Oznamovací balíky jsou na výstupu v čitelném tvaru. Měli byste v něm najít svoje konfigurační hodnoty; nenajdete-li je, oznámení pravděpodobně neposílá váš radvd ... vyhledejte na spojení jiný směrovač (a použijte LLAddress, což je fyzická adresa pro sledování).

Server DHCPv6 (dhcp6s)

DHCPv6 lze využít pro stavovou konfiguraci. Démon samotný nemusí běžet pouze na implicitním směrovači brány IPv6. Můžete uvést více informací než v případě radvd. Server je velmi podobný serveru DHCP IPv4. Po správném zkonfigurování démon reaguje na obdržené vícesměrové pakety ICMPv6 odeslané klientem na adresu ff02::16.

Konfigurace serveru DHCPv6 (dhcp6s)

Jednoduchá konfigurace

Konfigurační soubor dhcp6s se obvykle jmenuje /etc/dhcp6s.conf. Zde je jednoduchý příklad: interface eth0 { server-preference 255; renew-time 60; rebind-time 90;

```
prefer-life-time 130;
valid-life-time 200;
allow rapid-commit;
option dns_servers 2001:db8:0:f101::1 sub.domain.example;
link AAA {

range 2001:db8:0:f101::1000 to 2001:db8:0:f101::ffff/64; prefix
2001:db8:0:f101::/64; }; };
```

Konfigurace klienta DHCPv6 (dhcp6c)

Jednoduchá konfigurace

Konfigurační soubor dhcp6s se obvykle jmenuje /etc/dhcp6s.conf. Zde je jednoduchý příklad: interface eth0 { send rapid-commit; request domain-name-servers; };

Použití

dhcpv6_server

Start serveru, tj.

```
# service dhcp6s start
```

dhcpv6_client

Start klienta na pozadí, tj.

```
# dhcp6c -f eth0
```

Ladění

dhcpv6_server

Server má jeden klíč pro popředí a dva pro ladění (všechny by se měly používat pouze při ladění):

```
# dhcp6c -d -D -f eth0
```

dhcpv6_client

Klient má jeden klíč pro popředí a dva pro ladění, viz příklad:

```
# dhcp6c -d -f eth0 Oct/03/2005 17:18:16 dhcpv6 doesn't support hardware type 776 Oct/03/2005 17:18:16 doesn't support sit0 address family 0 Oct/03/2005 17:18:16 netlink_recv_rtgenmsg error Oct/03/2005 17:18:16 netlink_recv_rtgenmsg error Oct/03/2005 17:18:17 status code for this address is: success Oct/03/2005 17:18:17 status code: success Oct/03/2005 17:18:17 netlink_recv_rtgenmsg error Oct/03/2005 17:18:17 netlink_recv_rtgenmsg error Oct/03/2005 17:18:17 assigned address 2001:db8:0:f101::1002 prefix len is not in any RAs prefix length using 64 bit instead Oct/03/2005 17:18:17 renew time 60, rebind time 9
```

Poznamenejme, že chybová zpráva „netlink“ nemá žádný význam.

tcp_wrapper

tcp_wrapper je knihovna, která slouží k ochraně libovolné služby před zneužitím.

Filtrovací funkce

tcp_wrapper můžete využívat pro:

Filtrování podle zdrojových adres (IPv4 nebo IPv6).
Filtrování podle uživatelů (na klientovi musí běžet démon ident).

Které programy využívají tcp_wrapper

Známy jsou tyto:

Všechny služby volané démonem xinetd (je-li xinetd zkompilován s knihovnou tcp_wrapper).
Démon sshd (je-li zkompilován s knihovnou tcp_wrapper).

Použití

tcp_wrapper je řízen dvěma soubory jménem /etc/hosts.allow a /etc/hosts.deny. Další informace viz:

```
$ man hosts.allow
```

Příklad souboru /etc/hosts.allow

V tomto souboru musí mít každá pozitivně filtrovaná služba (tj. spojení je přijato) vlastní řádek

sshd: 1.2.3. [3ffe:ffff:100:200::]/64 daytime-stream: 1.2.3. [3ffe:ffff:100:200::]/64

Poznámka: Vyskytují se ještě i vadné implementace, které obsahují následující nesprávný popis sítě IPv6: [3ffe:ffff:100:200::/64]. Doufejme, že budou brzy odstraněny.

Příklad souboru /etc/hosts.deny

Tento soubor obsahuje položky se všemi negativními filtry a obvykle by měl odmítat zbytek pomocí:

ALL: ALL

Je-li uzel příliš citlivý, můžete standardní řádek shora nahradit řádkem následujícím, což však může umožnit útok typu DoS (odmítnutí služby, přetížení poštovního programu a vyrovnávacího adresáře pro spooling), pokud je v krátkém čase uskutečněno příliš mnoho spojení. Pro tento případ je pravděpodobně vhodnější nástroj logwatch.

ALL: ALL: spawn (echo "Attempt from %h %a to %d at `date`" | tee -a /var/log/tcp.deny.log | mail root@localhost)

Logy

V závislosti na položce v konfiguračním souboru /etc/syslog.conf démona syslog zapisuje tcp_wra-pper obvykle do souboru /var/log/secure.

Odmítnuté spojení

Odmítnutí spojení IPv4 s démonem xinetd, který zajišťuje službu daytime, má za následek výpis podobný následujícímu příkladu:

Jan 2 20:40:44 gate xinetd-ipv6[12346]: FAIL: daytime-stream libwrap ¬ from=::ffff:1.2.3.4 Jan 2 20:32:06 gate xinetd-ipv6[12346]: FAIL: daytime-stream libwrap from=3ffe:ffff:100:200::212:34ff:fe12:3456

Odmítnutí spojení IPv4 s démonem sshd, který zajišťuje dvojí naslouchání, má za následek výpis podobný následujícímu příkladu:

Jan 2 20:24:17 gate sshd[12345]: refused connect from ::ffff:1.2.3.4 ¬ (::ffff:1.2.3.4) Jan 2 20:39:33 gate sshd[12345]: refused connect from 3ffe:ffff:100:200::212:34ff:fe12:3456 ¬ (3ffe:ffff:100:200::212:34ff:fe12:3456)

Povolené spojení

Povolené spojení IPv4 s démonem xinetd, který zajišťuje službu daytime, má za následek výpis podobný následujícímu příkladu:

Jan 2 20:37:50 gate xinetd-ipv6[12346]: START: daytime-stream pid=0 ¬ from=::ffff:1.2.3.4 Jan 2 20:37:56 gate xinetd-ipv6[12346]: START: daytime-stream pid=0 ¬ from=3ffe:ffff:100:200::212:34ff:fe12:3456

Povolené spojení IPv4 s démonem sshd, který zajišťuje dvojí naslouchání, má za následek výpis podobný následujícímu příkladu:

Jan 2 20:43:10 gate sshd[21975]: Accepted password for user from ::ffff:1.2.3.4 ¬ port 33381 ssh2 Jan 2 20:42:19 gate sshd[12345]: Accepted password for user ¬ from 3ffe:ffff:100:200::212:34ff:fe12:3456 port 33380 ssh2

vsftpd Naslouchání na adresách IPv6

Volbu pro naslouchání nastavte v konfiguračním souboru (obvykle je to /etc/vsftpd/vsftpd.conf) takto:

listen_ipv6=yes

Toť vše.

proftpd Naslouchání na adresách IPv6

Volbu pro naslouchání nastavte v konfiguračním souboru (obvykle je to /etc/vsftpd/ proftpd.conf) takto (avšak pozor, ne vše je v nastavení virtuálního počítače stoprocentně logic-ké):

<VirtualHost 192.0.2.1> ... Bind 2001:0DB8::1 ...
</VirtualHost>

Toť vše.

Ostatní démoni

Nyní je situace poměrně jednoduchá. Chcete-li naslouchat na IPv6, vyhledejte buď příslušnou volbu na příkazovém řádku nebo konfigurační hodnotu. Přečtěte si manuálové stránky démona nebo si projděte často kladené otázky. Může se stát, že z důvodu nedostatku podpory lze někdy démona svázat pouze s „libovolnou" adresou IPv6 (*any*, ::), nikoli s určitou adresou. Záleží na tom, jak daleko je programátor s implementací...

Další informace a URL

Programování (s použitím API)

S programováním IPv6 nemám žádné zkušenosti. Možná, že kapitolu doplní někdo jiný nebo budepřenesena do jiného návodu.Informace naleznete v RFC 2553 a na adresách:

http://www.ietf.org/internet-drafts/
http://jungla.dit.upm.es/~ecastro/IPv6-web/ipv6.html, aut. Eva M. Castro

Součinnost

Ve světě existují projekty, které se zabývají součinností různých operačních systémů vzhledem k implementaci vlastností IPv6. Zde jsou některé odkazy:

- TAHI Project, *http://www.tahi.org/*

Tištěné knihy, články, on-line časopisy (směs) Tištěné knihy (anglicky)

Cisco

Cisco Self-Study: Implementing IPv6 Networks (IPV6), Regis Desmeules. Cisco Press; ISBN 1587050862; 500 stran; 1. vydání (11. 4. 2003).
Configuring IPv6 with Cisco IOS, Sam Brown, Neal Chen, Robbie Harrell, Edgar, Jr. Parenti (redaktor), Eric Knipp (redaktor), Paul Fong (redaktor), 362 stran; Syngress Media Inc; ISBN 1928994849 (12. 7. 2002).

Obecné

IPv6 Essentials, Silvia Hagen, červenec 2002, O'Reilly. Obj. číslo: 1258, ISBN 0-5960-0125-8; 352 stran. ToC, Index, Sample Chapter etc.; O'Reilly Pressrelease.
IPv6: The New Internet Protocol. By Christian Huitema; Vydalo Prentice-Hall; ISBN 0138505055. Popis: Autorem knihy je Christian Huitema – člen InternetArchitecture Board. Kniha nabízí vynikající popis IPv6, odlišnosti od IPv4 a otázky spojené s vývojem. Zdroj:

http://www.cs.uu.nl/wais/html/na-dir/internet/tcp-ip/resource-list.html.

IPv6 Networks, Niles, Kitty (ISBN 0070248079); 550 stran. Vyšlo 05/01/1998.

- Implementing IPv6. Supporting the Next Generation Internet Protocols, P. E. Miller, Mark

A. Miller. Vydavatel: John Wiley & Sons; ISBN 0764545892; 2. vydání (15/03/2000); 402 stran.
Big Book of IPv6 Addressing Rfcs, Peter H. Salus (překlad); vyd. Morgan Kaufmann Publishers; 04/2000; 450 stran; ISBN 0126167702.
Understanding IPV6, Davies, Joseph; ISBN 0735612455. Vyšlo 05/01/2001; 350 stran. Understanding IPV6, Davies, Joseph; ISBN 0735612455. Vyšlo 13/11/2002; 544 stran.
Migrating to IPv6 – IPv6 in Practice, Marc Blanchet. Vydavatel: John Wiley & Sons; ISBN 0471498920; 1. vydání (11/2002); 368 stran.
Ipv6 Network Programming, Jun-ichiro Hagino; ISBN 1555583180.
Wireless boosting IPv6, Carolyn Duffy Marsan, 23/10/2000.
O'Reilly Network, klíčové slovo IPv6, 29 odkazů (28/01/2002).

Tištěné knihy (německy)

Technik der IP-Netze (TCP/IP incl. IPv6); vyd. Amazon.de; Anatol Badach, Erwin Hoffmann; Carl Hanser Verlag; München, Wien, 2001 ISBN 3-446-21501-8. Kap. 6: Protokol IPv6, s. 205-242. Kap. 7: Plug&Play – podpora IPv6; s. 243-276. Kap. 8: Migrace v IPv6, s. 277–294. Kap. 9.3.4: RIP protokolu IPv6 (RIPng), s. 349-351. Kap. 9.4.6: OSPF pro IPv6, s. 384-385. Komentář: nikoli zcela aktuální a nikoli zcela bez chyb.

- Internet-Sicherheit (Browser, Firewalls und Verschlüsselung); vyd. Amazon. Kai Fuhrberg.

2. uprav. vyd., vyšlo r. 2000, Carl Hanser Verlag, München, Wien, ISBN 3-446-21333-3. Kap.
2.3.1.4. IPv6, s. 18-22. Stručný přehled: RFC1825 – Security Association Konzept RFC1826
– IP authentication Header RFC1827 – IP Encapsulation Security Payload.
IPv6. Das neue Internet-Protokoll. Technik, Anwendung, Migration; vyd. Amazon Hans Peter Dittler. 2. upr. a doplň. vyd. 2002, ISBN 3-89864-149-X.
Das neue Internetprotokoll IPv6; vyd. Amazon Herbert; Wiese 2002;Carl Hanser Verlag, ISBN 3446216855.

Tištěné knihy (česky)

IPv6. Satrapa Pavel, Neokortex 2002; 238 stran; ISBN 80-86330-10-9. Patrně jediná česká kniha věnovaná výhradně IPv6.
TCP/IP v kostce, Pužmanová Rita; Kopp 2004; 608 stran; ISBN 80-7232-236-2. Podrobný popis protokolů TCPI/IP, zabývá se částečně i IPv6.

Články, eKnihy, on-line časopisy (směs)

Getting Connected with 6to4, *http://www.onlamp.com/pub/a/onlamp/2001/06/01/ipv6_ tutorial.html*, Huber Feyrer, 06/01/2001.
Transient Addressing for Related Processes: Improved Firewalling by Using IPv6 and Mul-tiple Addresses per Host; written by Peter M. Gleiz, Steven M. Bellovin (PC-PDF-Version: *http://www.securiteinfo.com/ebooks/pdf/tarp.pdf*; Palm-PDF-Version: *http://www.securite info.com/ebooks/palm/tarp.pdf*; PDB-Version: *http://www.securiteinfo.com/ebooks/ pdb/tarp.pdb*).

- IPv6, théorie et pratique (francouzsky); *http://www.oreilly.fr/catalogue/ipv6-3ed.html*;

3. vydání, 03/2002, O'Reilly; ISBN 2-84177-139-3.
IPSec (francouzsky), *http://www.securiteinfo.com/crypto/IPSec.shtml.*
Internetworking IPv6 with Cisco Routers, *http://www.ip6.com/index.html*, Silvano Gai, McGrawHill Italia, 1997. 13 kapitol a přílohy A-D si lze stáhnout jako dokumenty PDF.
Secure and Dynamic Tunnel Broker, *http://www.vermicelli.pasta.cs.uit.no/ipv6/ students/vegars/*, Vegar Skaerven Wang, Disetrační práce v IT; 02/06/2000; Faculty of Sci-ence, Dep.of Computer Science, University of Tromso, Norsko.
Aufbruch in die neue Welt – IPv6 in IPv4 Netzen, *http://www.old.netobjectdays.org/pdf/ 99/stja/doering.pdf.* Dipl. Ing. Ralf Döring, TU Illmenau, 1999.
Migration and Co-existence of IPv4 and IPv6 in Residential Networks, *http://www.csc.fi/~psavola/residential.html*; Pekka Savola, CSC/FUNET, 2002.

Vědecké publikace (abstrakty, bibliografie, on-line zdroje)

GEANT IPv6 Workplan, *http://www.ipv6.ac.uk/gtpv6/workplan.html.*
A simulation study on the performance of Mobile IPv6 in a WLAN-based cellular network, Perez Costa X.; Hartenstein H. Computer Networks, 09/2002, sv. 40, č. 1, s. 191–204 (14); Elsevier Science.
IPv6 Trials on UK Academic Networks: Bermuda Project; Aug. 2002; *http://www.ipv6.ac.uk/ bermuda2/*, Participants – Getting connected – Project deliverables – Network topology – Address assignments – Wireless IPv6 access – IPv6 migration – Project presentations – Internet 2 – Other IPv6 projects – IPv6 fora and standards Bermuda 2...
http://www.ipv6.ac.uk/, http://www.ipv6.ac.uk/.
A scalable parallel internet router that enables the QoS through merging ATM with IPv6. Song S. – Computer Communications, 01/05/2002, sv. 25, č. 7, s. 647–651 (5) – Elsevier Science.

- Linux IPv6: Which One to Deploy? Linux Journal, č. 96, s. 86, 88–90, 04/2002.
- An overview and analysis of mobile Internet protocols in cellular environments;

http://www.ingenta.com/isis/searching/ExpandSearch/ingenta?year_to=2002&year_from=1 997&date_type=range&title=IPv6&title_type=tka&database=1&newMatches=false&page-Start=1&index=3, Chao H-C. – Internet Research: Electronic Networking Applications and Policy, 24/10/2001, sv. 11, č. 5, s. 435–450 (16) – MCB University Press
IPv6 for Future Wireless NetworksToftegaard Nielsen T. – Wireless Personal Communicati-ons, June 2001, sv. 17, č. 2/3, s. 237–247 (11) – Kluwer Academic Publishers, Dordrecht, Nizozemí.
IPv6 at the University of Southampton, *http://www.ipv6.ecs.soton.ac.uk/*
Seamless Support for Mobile Internet Protocol Based Cellular Environments, Chao H-C.; Chu Y-M. – International Journal of Wireless Information Networks, 07/2001, sv. 8, č. 3, s. 133–153 (21) – Kluwer Academic/Plenum Publishers, New York, USA.
IPv6: The Solution for Future Universal Networks. Lecture Notes in Computer Science, sv. 1818, s. 82, 2000.
Modeling and performance analysis for IPv6 traffic with multiple QoS classes. Zhang L.; Zheng L. – Computer Communications; 09/2001, sv. 24, č. 15, s. 1626–1636 (11) – Elsevier Science.
Threshold-Based Registration (TBR) in Mobile IPv6. Lecture Notes in Computer Science, sv. 1818, s. 150, 2000.
IPv6 Performance Analysis on FreeBSD Workstation Using Simple Applications. Lecture Notes in Computer Science, sv. 1961, s. 33, 2000.
Microsoft Research IPv6 Implementation (MSRIPv6): *http://www.research.microsoft.com/ msripv6/.*
New fr, *http://www.ingenta.com/isis/searching/ExpandSearch/ingenta?year_to=2002&year from=1997&date_type=range&title=IPv6&title_type=tka&database=1&newMatches=false&p ageStart=1&index=9.*

On-line informace Globální rejstříky

http://www.6bone.net/, http://www.6bone.net/6bone_hookup.html, http://ww.w.join. uni-muenster.de/6bone/6bone-teilnahme.html (německy),
http://www.join.uni-muenster.de/6bone/6bone-participation.html (anglicky).

Hlavní regionální rejstříky

Amerika: ARIN, *http://www.arin.net/.*
Evropa a Blízký východ: Ripe NCC, *http://www.ripe.net/.*
Asie/Pacifik: APNIC, *http://www.apnic.net/.*
Latinská Amerika a Karibik: LACNIC, *http://lacnic.org/.*
Afrika: AfriNIC, *http://www.afrinic.org/.*

Seznam hlavních (délka prefixu 32) umístění lokálních registrů je zde: *http://www.ripe.net/ ripencc/mem-services/registration/ipv6/ipv6allocs.html.*

Tuneloví makléři

Sourcecode, *http://www.vermicelli.pasta.cs.uit.no/ipv6/students/vegars/TunnelBroker/*; viz Vermicellis, disertační práce o tunelových makléřích, University of Tromso, Norsko.
Původní IPng. Tunelový makléř a zdroje IPv6, nyní migrovalo do *http://www.sixxs.net/main/.*
Eckesovy stránky: IPv6-with-Linux, *http://sites.inka.de/lina/linux/ipv6.html.*
tunnelc – tunelový klientský skript založený na perl: freshmeat.net; *http://freshmeat.net/ projects/tunnelc*, SourceForge: *http://sourceforge.net/projects/tunnelc*
Linux Advanced Routing & Traffic Control HOWTO, *http://howtos.linuxbroker.com/*

howtoreader.shtml?file=Adv-Routing-HOWTO.html#LARTC.TUNNEL-IPV6.ADDRESSING. Další informace a URL na *http://www.ipv6-net.de/.*

6to4

NSayer's 6to4 information, *http://www.kfu.com/~nsayer/6to4/.*
RFC 3068.

ISATAP

- *http://www.join.uni-muenster.de/Dokumente/Howtos/Howto_ISATAP.php?lang=en*, autor *http://www.join.uni-muenster.de/.*

Novinky a URL ostatních dokumentů

Náměty uvítáme!

http://www.ipv6-net.de/, německé fórum.
http://www.estoile.com/links/ipv6, aut. Anil Edathara.

Odkazy na protokoly

RFC/IPv6

Rozsah tohoto dokumentu neumožňuje publikování seznamu RFC/IPv6, avšak následující URL vás k tomuto seznamu přivedou:

Setříděný seznam je v *http://playground.sun.com/pub/ipng/html/specs/standards.html* nebo v *http://playground.sun.com/pub/ipng/html/specs/specifications.html*, aut. Robert Hinden.
http://www.ipv6.org/specs.html na IPv6.org.

Návrhy pracovních skupin

Návrhy pracovních skupin, které se týkají IPv6, naleznete na:

http://www.ietf.org/ids.by.wg/ipv6.html
http://www.ietf.org/ids.by.wg/ngtrans.html
http://www.ietf.org/ids.by.wg/dhc.html
http://www.ietf.org/ids.by.wg/dnsext.html
http://www.ietf.org/ids.by.wg/mobileip.html
http://playground.sun.com/pub/ipng/html/ipng-main.html (včetně dostupnosti zásobníků na různých platformách & zdrojový kód pro zásobníky IPv6)

Ostatní

http://www.networksorcery.com/enp/protocol/ipv6.htm, hlavička protokolu IPv6
http://www.switch.ch/lan/ipv6/references.html, velký seznam odkazů na IPv6, který udržu-je Simon Leinen.

Další informace

http://www.deepspace6.net/sections/links.html

Vztahující se k určité distribuci Linuxu

PLD

http://www.pld-linux.org/ („jednička na trhu“ v balících IPv6)

Red Hat

http://www.redhat.com/, *http://www.netcore.fi/pekkas/linux/ipv6/* (Pekka Savola)

Debian

http://www.debian.org/, *http://people.debian.org/~csmall/ipv6/*, *http://www.jipo.org/jim/Jims_LAN_IPv6_global_connectivity_howto.html*

Novell/openSUSE

http://www.novell.com/linux/suse/

Mandriva Linux

http://www.mandriva.com/ Další na stránkách *http://www.bieringer.de/linux/IPv6/status/IPv6+Linux-status-distributions.html.*

Obecně

http://www.ipv6.org/
http://www.6bone.net/
http://www.cs-ipv6.lancs.ac.uk/ – Spojené království
http://www.ipv6forum.com/ – světové konsorcium internetových prodejců, Research & Education Networks
http://playground.sun.com/pub/ipng/html/ipng-main.html – udržuje Robert Hinden, Nokia. Veškeré informace o IPv6, od přehledů přes RFC & návrhy až k implementacím (včetně dostupnosti zásobníků na různých platformách & zdrojový kód pro zásobníky IPv6)
http://www.6init.com/ – Internetová iniciativa IPv6 – Pátý rámcový program EU, součást programu IST..
http://www.ipv6-taskforce.org/
http://www.6init.org/ – IPv6 INternet IniTiative
http://www.isoc.org/HMP/PAPER/PT1/html/pt1.html.hinden
http://www.usenix.org/publications/library/proceedings/ana97/summaries/deering.html, aut. Steve Deering
http://www.garykessler.net/library/ipv6_exp.html, aut. Gary C. Kessler
http://www.3com.com/nsc/ipv6.html – 3Com
http://www.ngi.gov/
http://www.internet2.org/ a *http://ipv6.internet2.edu/ – http://ipv6.internet2.edu/ presentations/* z dílen IPv6 (Bezstavová autokonfigurace, adresování IPv6, USAGI, adreso-vání IPv6 nezávislé na poskytovateli a další témata)
NetworkWorldFusion (102 dokumentů k datu 22. 12. 2002)
http://www.theregister.co.uk/ (30 dokumentů IPv6, 22. 12. 2002)
http://zdnet.search.com/search?cat=279&q=IPv6
http://whatis.techtarget.com/wsearchResults/1,290214,sid9,00.html?query=IPv6
http://www.faqs.org/faqs/internet/tcp-ip/resource-list/index.html
http://ipv6.klingon.nl/, http://www.ipv6.klingon.nl/: příklady firewallů IPv6, testování šířky pásma a portscanner

Informace v češtině

http://www.ipv6.cz/
http://www.cesnet.cz/ipv6/
http://www.logix.cz/michal/doc/article.xp/ipv6-1

Bezpečnost IPv6

Internet Security Systems: Security Center, *http://www.iss.net/security_center/search.php? type=3&type=3&pattern=IPv6* (21. 12. 2002 – nalezeno 6 částí věnovaných IPv6)
http://csrc.nist.gov/ipsec/ (National Institute of Standards and Technology, NIST)
http://www.infosecuritymag.com/index.shtml
http://neworder.box.sk/search.php3?srch=IPv6 (články, využití, soubory, databáze atd.)

Seznamy aplikací

http://www.deepspace6.net/docs/ipv6_status_page_apps.html
http://www.ipv6.org/v6-apps.html
http://freshmeat.net/search/?q=IPv6, v současnosti (14/12/2002) 62 projektů
Fórum IPv6: *http://www.ipv6forum.com/navbar/links/v6routerlist.htm*

Nástroje na analýzu

http://www.wireshark.org/ – Wireshark (dříve Ethereal) je volně dostupný analyzátor pro-tokolů v systémech Unix a Windows.
http://www.ip6.com/us/analyzer.htm. Stáhnout lze: Radcom RC100-WL protocol analyzer, verze 3.20.

Produkty IPv6

http://www.6wind.com/ – řešení pro směrovače IPv4/IPv6, QoS, vícesměr, mobilitu, bez-pečnost/VPN/firewally.

http://www.fefe.de/dns/ 08/2002 – Co je djbdns a proč potřebuje IPv6? (djbdns je plnohod-notný DNS server, který předčí BIND prakticky ve všech ohledech)
http://www.ipinfusion.com/products/server/products_server.html.
Inframail (Advantage Server Edition) 6.0, *http://download.com.com/3000-2165-8202652.html?tag=lst-0-2.*
http://download.com.com/3000-2377-10149393.html?tag=lst-0-1.
ftp://ftp.porcupine.org/pub/ipv6/.

SNMP

- *http://www.cs.uu.nl/wais/html/na-dir/snmp-faq/part1.html*

Infrastruktura IPv6 Statistiky

http://www.space.net/~gert/RIPE/, aut. Gert Döring
http://6bone.informatik.uni-leipzig.de/ipv6/stats/stats.php3
http://www.ripe.net/ripe/meetings/archive/ripe-42/presentations/ripe42-ipv6-survey/sld001.html, IPv6 WG, Ripe 42, Ripe NCC

Propojovací centra

Seznam propojovacích center lze nalézt na: *http://www.v6nap.net/* nebo na *http://www.euro-ix.net/isp/choosing/search/matrix.php.*

Evropa

- *http://www.euro6ix.net/*, European IPv6 Internet Exchange Backbone

USA

http://www.6tap.net/: Chicago. Supports peerings around the globe
http://www.ny6ix.net/: New York City IPv6 based Internet Exchange
http://www.paix.net/: Palo Alto

Tuneloví makléři

Viz také: *http://www.deepspace6.net/docs/tunnelbrokers.html*
Evropa

- *http://www.xs26.net/*, USA & Evropa

Německo

http://6bone.informatik.uni-leipzig.de/
http://fix.ipv6.berkom.de/cgi-bin/tb.pl

USA

http://www.es.net/hypertext/welcome/pr/ipv6.html, USA – Energy Sciences Network: Tun-nel Registry & Address Delegation for directly connected ESnet sites and ESnet collabo-rators.
http://www.6ren.net/, USA – Iniciativa 6ren je koordinovaná společností Energy Sciences Network (ESnet), což je síť pro energetický výzkumný program Ministerstva pro energii USA, je umístěna v kalifornské laboratoři Lawrence Berkeley National Laboratory.
http://www.xs26.net/, USA & Evropa
http://ipv6tb.he.net/, US backbone; *http://tunnelbroker.com/*

Další tuneloví makléři...

- *http://www.kfu.com/~nsayer/6to4/* (bojkot MS IIE!)

Služby IPv6

Poznámka: Tyto služby jsou většinou dostupné pouze s platným připojením IPv6!

Novinky na síti (NNTP)

- *nntp://news.ipv6.scarlet-internet.nl/* (dostupné přes všechny SixXS POPy).

Server s hrami

- *http://www.viagenie.qc.ca/en/ipv6/quake2/ipv6-quake2.shtml* přes IPv6.

IRC Server

- *http://ipv6.cyconet.org/?id=server* (Cyconet IRCnet Servery prostřednictvím IPv6).

Rozhlasové stanice, hudební kanály

http://aopteryx.informatik.uni-leipzig.de:8000/live.mp3, Lipská univerzita, Německo.
http://www.ipv6.bieringer.de/.

www server

Ještě něco chybí? Náměty uvítáme!

E-mailové konference

Seznam konferencí naleznete na:

- *http://www.deepspace6.net/sections/lists.html.*

Hlavní konference jsou uvedeny v následující tabulce:

Téma	e-mailová adresa	Co lze vybrat	Adresa konference	Jazyk	Přístup přes WWW
Jádro Linuxu, sítě včetně IPv6	majordomo (at) oss.sgi.com	netdev	netdev@oss.sgi.com	anglicky	http://oss.sgi.com/projects /netdev/archive/
Linux a IPv6 obecně	majordomo (at) list.f00f.org	linux-ipv6	linux-ipv6@list.f00f.org (moderovaná)	anglicky	(1)
Linuxové implementace protokolu IPv6	Web-based, see URL		project6@ferrara.linux.it	anglicky	http://project6.ferrara.linux.it / sections/lists.html
Mobilní IP(v6) v Linuxu	majordomo (at) list.mipl.mediapoli.com	mipl	mipl@list.mipl.mediapoli.com	anglicky	http://www.mipl.mediapoli.com / mailinglist.html
Linuxoví uživatelé rozšíření IPv6 USAGI	majordomo (at) linux-ipv6.org	usagi-users	usagi-users@linux-ipv6.org	anglicky	http://www.linux ipv6.org/ml /index.html# usagi-users
IPv6 v Debian Linuxu	Web-based, see URL		debian-ipv6@lists.debian.org	anglicky	http://lists.debian.org/ debian-ipv6/
IPv6/6bone v Německu	majordomo (at) atlan.uni-muenster.de	ipv6	ipv6@uni-muenster.de	německy / anglicky	http://www.join.uni-muenster.de/ JOIN/ipv6/texte - englisch/mailinglis te.html
6bone	majordomo (at) isi.edu	6bone	6bone@isi.edu	anglicky	http://www.6bone.net /6bone_email.html
Diskuse o IPv6	majordomo (at) sunroof.eng.sun.com	ipng	ipng@sunroof.eng.sun.com	anglicky	http://playground.sun.com/pub /ipng/html/instructi

				ons.html
Obecně o uživatelích majordomo (at) IPv6 ipv6.org	users	users@ipv6.org	anglicky	http://www.ipv6.org/mailing-lists.html
Ladění internetových aplikací (2)	bugtraq-subscribe (at) securityfocus.com	bugtraq@securityfocus.com (moderated)	anglicky	http://online.securityfocus.com/popups/forums/bugtraq/intro.shtml
Obecně o IPv6	Web-based, see URL	ipv6@ipng.nl	anglicky	http://mailman.ipng.nl/mailman/listinfo/ipv6/
majordomo (at) mfa.eti.br	majordomo (at) mfa.eti.br	ipv6@mfa.eti.br	portugalsky	http://www.marcelo.pro.br /mailman/listinfo/ipv6

(1) vhodné pro běžné revize Linuxu & IPv6

(2) velmi vhodné pro serverové aplikace

On-line nástroje Nástroje na testování

finger, nslookup, ping, traceroute, whois: *http://www.cs-ipv6.lancs.ac.uk/ipv6/testing/.*
ping, traceroute, tracepath, 6bone registry, DNS: *http://www.join. uni-muenster.de/lab/testtools.html* (pouze v němčině, avšak budou rozumět i ti, kdo neumí německy).
traceroute6, whois: *http://www.ipng.nl/.*
AAAA Lookup Checker *http://www.cnri.dit.ie/cgi-bin/check_aaaa.pl.*
Různé nástroje: *http://www.ipv6tools.com/.*
http://doc.tavian.com/ipv6util/index.htm (podobné ipv6calc).

Získávání informací

http://www.kessens.com/~david/6bone/,
http://www.ripe.net/ripencc/mem-services/registration/ipv6/ipv6allocs.html.

Zrcadla IPv6

http://linux.uninet.edu/lg/,
http://www.v6.dren.net/lg/.

Pomocné aplikace

http://www.tdoi.org/prefcalc.php, aut. *http://www.tdoi.org/*,
http://www.maths.tcd.ie/cgi-bin/check_dns.pl.

Školení, semináře

http://www.aerasec.de/workshops/ipv6.html, AERAsec, Německo (prozatím pouze v němči-ně).
http://www.seminarinformation.com/wconnect/wc.dll?sis~details0~194045, Learning Tree International.
http://www.e-trainonline.com/html/ciw_internetworking_profession.html#IPv6, CIW Inter-networking Professional Training CBT CD.
http://www.trainingpages.net/x/category.html?kw=125, U.K. (13 kurzů, 22. 12. 2002). Chybí něco? Náměty uvítáme!

„Výzkum on-line“

IPv6: Addressing The Needs Of the Future, *http://www.amazon.com/exec/obidos/tg/detail/-/B00006334Y/copernicshopper/103-1420916-1341420* [KE STAŽENÍ V PDF], Yankee Group (autor). Cena: $595.00. Vydání: e-book (Acrobat Reader). Počet stran: 3 (tři). Vydavatel: Marke-tResearch.com; ISBN B00006334Y (1/11/2001).

Zajímavý by mohl být počet výtisků...

Stručný úvod do IRC

Úvod

Informace týkající se probíraných témat naleznete i v následujících příručkách:

Dokument RFC 1459 (Jarkko Oikarinen a Darren Reed) se jako první věnoval protokolu Internet Relay Chat Protocol – *http://ftp.isi.edu/in-notes/rfc1459.txt*
RFC 2810 (Christophe Kalt) aktualizuje RFC 1459 a popisuje architekturu Internet Relay Chat – *http://ftp.isi.edu/in-notes/rfc2810.txt*
RFC 2811 (Christophe Kalt) aktualizuje RFC 1459 a popisuje správu kanálů IRC –

http://ftp.isi.edu/in-notes/rfc2811.txt

- RFC 2812 (Christophe Kalt) aktualizuje RFC 1459 a popisuje klientský protokol IRC – *http://ftp.isi.edu/in-notes/rfc2812.txt*
- RFC 2813 (Christophe Kalt) aktualizuje RFC 1459 a popisuje serverový protokol IRC –

http://ftp.isi.edu/in-notes/rfc2813.txt Navštívit byste měli i web

http://www.irchelp.org/.

Cíle

Kromě jiného jsou cíle tohoto dokumentu následující:

Seznámit se s odkazy na důležité zdroje, které se zabývají IRC
Předejít častým chybám při používání IRC
Seznámit se s oblíbenými klienty, servery, roboty (bots), proxy bránami (bouncers) a jejich správci, s IRC #kanály, stručným popisem, možností stažení, domovskými stránkami a radami
Seznámit se s dostupnými nástroji IRC

O IRC

Výňatek z RFC2810:

Protokol IRC (Internet Relay Chat) se používá při textově založených konferencích. Vyvíjí se od roku 1989, kdy byl původně implementován jako prostředek pro uživatele na BBS, kteří mezi sebou chatovali.

Poprvé protokol formálně zdokumentovala v květnu roku 1993 příručka RFC 1459 [IRC] a od té doby se neustále rozvíjí. Protokol IRC je založen na modelu klient/server a je vhodný pro použití u množství počítačů v různých umístěních. Typická instalace představuje jediný krok (server), při němž se vytvoří cent-rální bod pro klienty (či jiné servery), ke kterým se připojuje a probíhá doručování zpráv/více-směrové vysílání a další funkce.

Tento model, který vyžaduje, aby každý server měl kopii informací o globálním stavu, je stále největším problémem protokolu, protože se tím omezuje maximální velikost, jaké může síť dosáhnout. Jestliže se existující sítě mohly dostatečně rozrůstat, musíme poděkovat výrobcům hardwaru, že nám dodávají stále výkonnější systémy.

Stručná historie IRC

První démon pro IRC vytvořil v polovině roku 1998 Jarkko „WiZ“ Oikarinen z univerzity v Oulu,Finsko. Původně byl vytvořen podle BBS jako náhrada programu Talk a rychle se rozšiřoval; nej-dříve ve Skandinávii a následně i ve zbytku světa. Během jednoho roku existovalo přes 40 pro-pojených serverů.

V dané fázi existovala pouze jedna síť, proto nebylo nutné ji pojmenovat – jmenovala se jedno-duše „IRC“; s rozrůstáním sítě však začalo docházet k nedorozuměním. IRC byl dosti chaotickýmmédiem s rozdělenými sítěmi, kolizemi nicků a různými propojeními komunikačních kanálů; bylonevyhnutelné, aby se uživatelé v určité fázi rozhodli vytvořit své vlastní sítě.

K jednomu z prvních podstatných rozdělení došlo v roce 1992, kdy Wildthang vytvořil síť Under-net. Původně mělo jít o

testovací síť, ale síť Undernet se rychle rozrůstala a díky zavádění služebpro ochranu uživatelů a komunikačních kanálů získala pověst přátelské sítě.

O dva roky později se rozdělila i síť Undernet a vznikla nová síť s názvem DALnet. Zakladatelemsítě DALnet byl dalvenjah, který povýšil princip služeb sítě Undernet na novou úroveň a zavedlregistraci nicků, linky G-lines a nabízel i další funkce.

Mezitím se změnily i názory v síti IRCnet (pod tímto názvem byla nyní známá síť IRC). Síť IRCnetvzdorovala principu „vlastnictví komunikačního kanálu/nicku“, který zavedly sítě Underneta DALnet, bylo však nutné řešit neustálé protínání komunikačních kanálů. Představeny byly dvěrůzné možnosti: prodleva nicku/komunikačního kanálu a časová razítka (informace naleznete nawebu *http://www.irc-help.org*), došlo však k rozhořčené debatě o tom, kterou z možností imple-mentovat.

V červenci roku 1996 se síť IRCnet rozdělila, přičemž se většina severoamerických serverů zfor

movala do sítě EFnet a síť IRCnet tak zůstala převážně evropská.Od té doby vznikly stovky dalších menších sítí, z nichž většina používá upravené verze DALnet,EFnet, IRCnet či Undernet.

Začínáme používat IRC

Standardním klientem IRC je původní ircII. Je součástí většiny distribučních balíčků systému Linux a je od něj odvozena většina ostatních textově založených klientů IRC (zvláště BitchX a EPIC).

Spuštění programu ircII

Používání klienta ircII je snadné. Řekněme například, že se chcete připojit k irc.freenode.net jako uživatel mini-HOWTO.
Do příkazového řádku napište:

```
$ irc mini-HOWTO irc.freenode.net
```

Můžete také exportovat proměnné, abyste je nemuseli používat v příkazovém řádku. Pro uživate-le bash a zsh:

```
$ export IRCNICK=mini-HOWTO IRCSERVER=irc.freenode.net
```

Uživatelé csh a tcsh uvedou setenv místo export.
Po dokončení přidejte proměnné do svého profilu shell (například ~/.bash_profile či ~/.zpro-file, viz část „Úvod do Linuxu“, kapitolu „Konfigurační soubory shellu“). Dalšími běžnými proměnnými jsou IRCNAME a IRCUSER pro nastavení části ircname v /whois a uži-
vatelské jméno v prvním řádku „mini-HOWTO ~username@hostname (ircname)“. Nezapomínejte, že IRCUSER nebude fungovat, pokud používáte ident daemon (který je ve většině distribučních balíčků výchozí). Potřebujete-li změnit své uživatelské jméno (nedoporučuje se, abyste používali IRC při přihlášení jako root!), nainstalujte oidentd z webu *http://ojnk.sourceforge.net/*. Informace pro konfiguraci naleznete v dokumentu oidentd.conf. Nakonec spusťte:

```
# /usr/local/sbin/oidentd -g nobody -u nobody
```

Po dokončení přidejte příkaz do svých spouštěcích skriptů (například /etc/rc.d/rc.local). Pokud nejsou proměnné IRCNICK, IRCUSER a IRCNAME nastaveny, budou získány z /etc/passwd (čili z informací o uživateli).

Příkazy

Pro zobrazení seznamu všech dostupných příkazů použijte /help (nápověda /help je dobrý začá-tek). Nahraďte *nick* jakýmkoli IRCNICK.

Nejdříve vypněte /set NOVICE.
/nick IRC-mini-HOWTO změní proměnnou IRCNICK na *IRC-mini-HOWTO*.
/set realname The Linux IRC mini-HOWTO změní proměnnou IRCNAME na *The Linux IRC mini-HOWTO* (nemění se u stejného připojení).
/j #mini-HOWTO se připojí ke komunikačnímu kanálu *#mini-HOWTO*.
/j #unmaintained-HOWTO se připojí ke komunikačnímu kanálu *#unmaintained-HOWTO*.
/j #mini-HOWTO změní aktuální aktivní komunikační kanál na *#mini-HOWTO*.
/msg *nick* Ahoj odešle soukromou zprávu pro *nick* obsahující *Ahoj*.
/notice *nick* (či #mini-HOWTO) Ahoj. odešle oznámení pro *nick* (či #mini-HOWTO) obsa-hující *Ahoj*.
/query *nick* spustí soukromou konverzaci s *nickem*; /query ukončí soukromou konverzaci.
/me uses *Linux* odešle akci do aktuálního komunikačního kanálu či dotaz obsahující *IRC-mini-HOWTO loves Linux*.
/dcc chat *nick* zahájí chat s *nickem*. Použijte /msg =*nick* (všimněte si znaku =) pro odesí-lání zpráv prostřednictvím chatu.
/dcc send *nick* /etc/HOSTNAME odešle daný soubor *nicku*.
/dcc get *nick* přijme soubor nabízený *nickem*.
/part opustí aktuální aktivní komunikační kanál.
/part #unmaintained-HOWTO opustí kanál *#unmaintained-HOWTO*.

/discon se odpojí od aktuálního IRCSERVER.
/server irc.oftc.net připojí k IRCSERVER *irc.oftc.net.*
/quit Bye ukončí vaši relaci IRC s důvodem *Bye.*

Většina z výše uvedených příkazů (včetně použití proměnných prostředí) bude fungovat také v jiných textově založených klientech.

Etiketa služby IRC

VAROVÁNÍ

- Nikdy nepoužívejte IRC, jste-li přihlášení jako root či jako kterýkoli jiný uživatel s rozšíře-nými oprávněními. Dříve či později se něco stane. Varovali jsme vás. Velmi doporučujeme, abyste vytvořili uživatele, který bude sloužit pouze k používání služby IRC. Viz:

  ```
  $ man adduser
  ```

V komunikačních kanálech Linux byste neměli:

Chovat se jako idiot. Chcete-li být respektováni, respektujte všechny ostatní.
Používat barvy (^C). Většina uživatelů systému Linux takové blázniviny pocházející z programu mIRC netoleruje a ircII je ve skutečnosti nepodporuje. To samé platí pro ANSI.
Používat pouze velká písmena (CAPS), tučné písmo (^B), reverzní znaky (^V), podtržení (^_), blikání (^F) a zvonění (^G). První 4 slouží ke zdůraznění slov, ne celého textu. Poslední 2 jsou jednoduše otravné.
Ptát se, zda se můžete na něco zeptat. Prostě se zeptejte, předtím si však přečtěte veš-kerou dostupnou dokumentaci týkající se daného tématu. Začněte s /usr/share/doc (v některých systémech může jít o /usr/doc), případně navštivte web *http://www.tldp.org/* nebo *http://www.ibiblio.org/pub/Linux/docs/.* Neopakujte své otázky několikrát za sebou. Počkejte alespoň 10 minut. Nedostanete-li žádnou odpověď, je to proto, že vám nikdo nemůže nebo nechce pomoci. Respektujte jejich volbu, nejsou vaši-mi osobními asistenty. Nikdy také neposílejte příliš mnoho soukromých zpráv. Je to stejné jako se spamem.

Klienti IRC pro textový režim

ircII

Správce: ircII project.*Komunikační kanál IRC:* #ircII (oficiální komunikační kanál) v síti EFNet *http://www.efnet.org/?module=servers.*

ircII vytvořil Michael Sandrof a je k dispozici ve většině distribucí Linuxu. Používá termcap a není vhodný pro většinu uživatelů, jedná se však o standard. Používat jej bude Mathusalem a další odborníci. Méně zkušení budou po jeho instalaci litovat.

Aktuální verzi ircII získáte na webu ftp://ircii.warped.com/pub/ircII/. Domovskou stránku nalez-nete na internetové adrese *http://www.eterna.com.au/ircii/.*

EPIC

Správce: EPIC Software Labs.*Komunikační kanál IRC:* #EPIC v síti EFNet.
Klient EPIC (Enhanced Programmable ircII Client) je založen na ircII a je určen pro skutečné tvůr-ce skriptů a uživatele, kteří hledají svobodu. Při jeho prvním spuštění zjistíte, že byste se skuteč-ně měli naučit základy skriptování.

Aktuální verzi EPIC získáte na webu *http://prbh.org/?page=ftp.* Domovskou stránku naleznete na internetové adrese *http://www.epicsol.org/.*

BitchX

Správce: Colten Edwards.*Komunikační kanál IRC:* #BitchX v síti EFNet.Původně se jednalo o skript pro ircII, nyní je však

BitchX oblíbeným klientem, který snižuje potře-
bu vytváření skriptů, protože nabízí obrovské množství funkcí v klientovi jako takovém (všechny dostupné funkce si nezapamatují ani zkušení uživatelé). BitchX bývá považován za přeceňovaný software, mezi uživateli je však velmi oblíbený.

Aktuální verzi BitchX získáte na webu *http://www.bitchx.org/download.php.* Domovskou stránkunaleznete na internetové adrese *http://www.bitchx.org/.* BitchX pravděpodobně najdete i ve svédistribuci Linuxu.

irssi

Správce: Timo Sirainen.
Komunikační kanál IRC: #irssi na webu freenode (*http://freenode.net/irc_servers.shtml*) a IRCnet (*http://www.ircnet.org/*). Timo

zveřejnil v roce 1997 klienta yagIRC. Jednalo se o grafického klienta, který využíval sadu nástrojů GTK+. Rok poté byl povolán na vojnu a nikdo nepřevzal správu klienta yagIRC. Po návra-tu vytvořil klienta irssi jako náhradu za yagIRC. Používal GTK+. Později se objevily verze pro GNOME a textovou konzoli. Od verze 0.7.90 se jedná pouze o klienta ve standardním textovém režimu. Podporuje skriptování v jazyce Perl.

Aktuální verzi irssi získáte na webu *http://irssi.org/?page=download.* Domovskou stránku nalezne-te na internetové adrese *http://irssi.org/.*

Další textoví klienti IRC

Existuje několik dalších klientů založených na ircII:

Blackened (*ftp://ftp.blackened.com/pub/irc/blackened/*).
Ninja *(ftp://qoop.org/ninja/*).
ScrollZ (*http://www.scrollz.com/*).

Klienti IRC pro X Window

Zircon

Správce: Lindsay F. Marshall.*Komunikační kanál IRC:* Žádný?Vytvořen v Tcl/Tk, používá nativní komunikaci Tcl.Aktuální verzi Zircon získáte na webu ftp://catless.ncl.ac.uk/pub/. Domovskou stránku naleznete na internetové adrese *http://catless.ncl.ac.uk/Programs/Zircon/.*

KVIrc

Správce: Szymon Stefanek.*Komunikační kanál IRC:* #KVIrc na webu freenode.Tento klient je vytvořen také v knihovně Qt, přičemž je velmi výkonný. Podporuje DCC Voice, vestavěný skriptovací jazyk a moduly plug-in.Aktuální verzi KVIrc získáte na webu *http://www.kvirc.net/?id=download.* Domovskou stránkunaleznete na internetové adrese *http://www.kvirc.net/.*

X-Chat

Správce: Peter Zelezny.*Komunikační kanál IRC:* #Linux na webu ChatJunkies – *http://www.chatjunkies.org/servers.php.*Používá GTK+, případně i GNOME, podporuje skriptování Perl a Python.Aktuální verzi X-Chat získáte na webu *http://xchat.org/download/.* Domovskou stránku naleznete na internetové adrese *http://xchat.org/.*

QuIRC

Správce: Patrick Earl.*Komunikační kanál IRC:* #QuIRC v síti DALnet – *http://www.dal.net/servers/index.php3.*Za použití Tk, podporuje Tcl pro skriptování.Aktuální verzi QuIRC získáte na webu *http://quirc.org/.*

Servery IRC

IRCD

Správce: vývojáři ircd.*Komunikační kanál IRC:* #ircd v síti IRCnet.Původní daemon IRC používaný hlavně v síti IRCnet. Novější verze se pokusily zdokonalit zabez-pečení komunikačních kanálů prostřednictvím dalších typů kanálů (například !linux) a režimů kanálů.Aktuální verzi IRCD získáte na webu ftp://ftp.irc.org/irc/server/. Domovskou stránku naleznete nainternetové adrese *http://www.irc.org/.*

IRCD-Hybrid

Správce: Neznámý?*Komunikační kanál IRC:* Žádný?Používaný hlavně v síti EFNet. Hybrid se zaměřuje na rychlost a výkon, přičemž mu chybí mnoho služeb nabízených v jiných sítích. Podle filozofie sítě EFnet, že se klienti nemají zabývat chodemkomunikačních kanálů, nepovoluje Hybrid klientům nastavovat režimy v kanálech či se připojo-vat ke kanálům, z kterých byli vyloučeni. Není tomu tak dávno, kdy Hybrid nabídl možnost, abyslužba znovu aktivovala kanál bez klientů.

Aktuální verzi IRCD-Hybrid získáte na webu *http://ftp1.sourceforge.net/ircd-hybrid/.* Domovskoustránku naleznete na internetové adrese *http://www.ircd-hybrid.org/.*

ircu

Správce: Undernet Coder Committee.*Komunikační kanál IRC Channel:* #ircu v síti Undernet (*http://www.undernet.org/servers.php*).Používá se převážně v síti Undernet. Po řadě útoků DDoS v letech 2001/2002 nabídl klient ircu uživatelům a serverům možnost skrývat své adresy. Ze sítě Undernet bylo také odstraněno mnoho příkazů /stats.Aktuální verzi ircu získáte na webu *http://ftp1.sourceforge.net/undernet-ircu/.* Domovskou stránkunaleznete na internetové adrese *http://coder-com.undernet.org/.*

Bahamut

Správce: DALnet Coding Team.*Komunikační kanál IRC:* #Bahamut v síti DALnet.Bahamut je server DALnet založený na DreamForge a Hybrid. Mezi funkce patří registrace komunikačních kanálů a nicků, podpora pro délku nicků až 15 znaků a evidenční služba.Aktuální verzi Bahamut získáte na webu *http://bahamut.dal.net/download.php.* Domovskou strán-ku naleznete na internetové adrese *http://bahamut.dal.net/.*

IRC „bots“

Eggdrop

Správce:Komunikační kanál IRC: #eggdrop v síti Undernet.Eggdrop nabízí silnou ochranu komunikačního kanálu a díky skriptování Tcl je velmi přizpůsobitelný.

Více eggdrop botů lze propojit a vytvořit tak celek, který zahrnuje větší počet komunikačních kanálů, či dokonce sítí. V sítích bez služeb registrace kanálů je eggdrop běžnou funkcí téměř všechvětších komunikačních kanálů.Aktuální verzi eggdrop získáte na webu *http://www.eggheads.org/downloads/.* Domovskou strán-ku naleznete na internetové adrese *http://www.eggheads.org/.*

EnergyMech

Správce: Proton.*Komunikační kanál IRC:* Žádný.První verze EnergyMech (či emech) si získala svou nepříliš příznivou pověst díky své schopnosti spouštět více chatů z jednoho procesu (což umožňovalo méně žádoucím uživatelům načíst stov-ky klonů z účtu). Takovou reputaci si však emech nezaslouží, protože ve skutečnosti nabízí dob-rou ochranu komunikačních kanálů a aktuální verze programu emech této funkci zamezily (namaximálně 4 instance). Jedná se o oblíbenou alternativu k Eggdrop.

Aktuální verzi EnergyMech získáte na webu *http://www.energymech.net/download.html.* Domov-skou stránku naleznete na internetové adrese *http://www.energymech.net/.*

IRC Bouncers (IRC Proxy)

bnc

Správce: James Seter.*Komunikační kanál IRC:* ŽádnýNástroj bnc je původní bouncer.Aktuální verzi bnc získáte na webu *http://gotbnc.com/download.html.* Domovskou stránku nalez-nete na internetové adrese *http://gotbnc.com/.*

muh

Správce: Sebastian Kienzl.*Komunikační kanál IRC:* ŽádnýNástroj muh je mnohostranný nástroj, který zamezuje zjišťování adresy IP a po spuštění bude brá-nit pokusům o získání vašeho nicku.Aktuální verzi nástroje muh získáte na webu *http://ftp1.sourceforge.net/muh/.* Domovskou stránkunaleznete na internetové adrese *http://seb.riot.org/muh/.*

ezbounce

Správce: Murat Deligönül.*Komunikační kanál IRC:* ŽádnýK základním funkcím ezbounce patří ochrana hesla, vzdálená správa, přihlašování a sledování vět-

šího množství portů.Aktuální verzi ezbounce získáte na webu *http://druglord.freelsd.org/ezbounce/*.

Instalace

Klienti

Všichni oblíbení klienti používají *GNU* Autoconf a *GNU* Automake, proto jsou k dispozici s konfi-guračním skriptem. Po rozbalení zdrojů si přečtěte pokyny k instalaci. Před provedením kompila-ce se ujistěte, zda máte požadované knihovny. Správným postupem je:

cd nazev_vytvoreneho_adresare # ./configure --help # ./configure --zde_jsou_vase_moznosti # make; # make install > ~/sources_install.log.

Pro ircII, EPIC a BitchX byste měli upravit include/config.h tak, aby odpovídal vašim potře-bám.

Servery

Skutečně potřebujete pomoc pro nastavení serveru?

~$ touch ircd.conf

Co je součástí mé distribuce? (Linux v počítačích x86)

11.1. Debian

IRC Channel: #Debian na webu freenode (irc.debian.org -> irc.freenode.net)Debian (*http://www.debian.org/*) nabízí tolik nástrojů IRC, že je nemůžeme všechny uvést. Nalez-nete je na následujících webech:

Stabilní Debian – *http://ftp.debian.org/debian/dists/stable/main/binary-i386/.*
Nestabilní Debian (neproběhlo dostatečné testování) – *http://ftp.debian.org/debian/ dists/unstable/main/binary-i386/.*
Plánované aktualizace naleznete na internetové adrese *http://ftp.debian.org/debian/ dists/proposed-updates/.* Může obsahovat také klienty IRC.

Fedora (Red Hat)

Komunikační kanál IRC: #RedHat na webu freenode (irc.redhat.com -> irc.freenode.net)Fedora Core 3 (*http://fedora.redhat.com/*) nabízí následující klienty (v novějších verzích najdeteaktualizované balíčky):

EPIC4 1.0.1 – *http://download.fedora.redhat.com/pub/fedora/linux/core/3/i386/os/ Fedora/RPMS/epic-1.0.1-18.i386.rpm.*
KSirc z KDE Network 3.3.0 – *http://download.fedora.redhat.com/pub/fedora/linux/core/3/ i386/os/Fedora/RPMS/kdenetwork-3.3.0-5.i386.rpm.*
X-Chat 2.4.0 – *http://download.fedora.redhat.com/pub/fedora/linux/core/3/i386/os/ Fedora/RPMS/xchat-2.4.0-3.i386.rpm.*
Rawhide (vývojová verze) – *http://download.fedora.redhat.com/pub/fedora/linux/core/ development/.* Používání je na vlastní nebezpečí.

Slackware

Komunikační kanál IRC: #Slackware na webu freenode a OFTC – *http://www.oftc.net/* Slackware (*http://www.slackware.com/*) verze 10.0 nabízí následující klienty:

BitchX 1.1 – *ftp://ftp.slackware.com/pub/slackware/slackware-10.0/slackware/n/bitchx-1.1-i486-1.tgz.*
EPIC4 2.0 – *ftp://ftp.slackware.com/pub/slackware/slackware-10.0/slackware/n/epic4-2.0-i486-1.tgz.*
irssi 0.8.9 – *ftp://ftp.slackware.com/pub/slackware/slackware-10.0/slackware/n/irssi-0.8.9-i486-3.tgz.*
KSirc od KDE Network 3.2.3 – *ftp://ftp.slackware.com/pub/slackware/slackware-10.0/slack-ware/kde/kdenetwork-3.2.3-i486-1.tgz.*
X-Chat 2.0.9 – *ftp://ftp.slackware.com/pub/slackware/slackware-10.0/slackware/gnome/ xchat-2.0.9-i486-1.tgz.*
Slackware – current (ve vývoji) – *ftp://ftp.slackware.com/pub/slackware/slackware-current/.* Používání je na vlastní nebezpečí.

Mandriva Linux

Mandriva Linux nabízí v distribučních zdrojích Main a Contrib následující balíčky (pro instalaci použijte urpmi):

BitchX,
irssi,

X-chat,
EPIC,
Ksirc.

LDAP v Linuxu

Úvod

V tomto dokumentu je popsána instalace, konfigurace, provozování a údržba serveru LDAP (Light-weight Directory Access Protocol) na linuxovém počítači. V dokumentu jsou rovněž uvedenypodrobnosti o vytváření databází LDAP a jsou zde popsány postupy při přidávání, aktualizacia rušení dat v adresáři. Tento návod vychází z informačních stránek LDAP University of Michigana z Průvodce správou OpenLDAP (OpenLDAP Administrator's Guide).

Autorem je Luiz Ernesto Pinheiro Malere, <*malere@yahoo.com*>.Hlavním cílem tohoto dokumentu je poskytnout návod, jak nastavit a používat adresářový serverLDAP na linuxovém počítači. Dozvíte se, jak se instaluje, provozuje a udržuje server LDAP. Dálese dozvíte, jak máte pomocí klientů a utilit LDAP ukládat, číst a aktualizovat data v adresáři.

Démon adresářového serveru LDAP se nazývá *slapd* a běží na mnoha různých unixových platfor-mách.Další démon zajišťuje replikace mezi servery LDAP. Nazývá se *slurpd*, avšak v tomto dokumentu
se jím nemusíme zabývat. Budeme používat pouze démona slapd, který zajišťuje adresářové služ-by pouze pro lokální doménu, bez replikací, takže i bez slurpd. Úplné informace o replikacíchjsou k dispozici na stránkách OpenLDAP Administrator's Guide (*http://www.openldap.org/ doc/-admin22/replication.html*).

Nastavení lokální domény představuje jednodušší variantu konfigurace serveru, s níž je vhodnézačít a kterou je možné v případě potřeby snadno později změnit. Informace uvedené v tomtodokumentu představují důležitý úvod k používání serveru LDAP. Je možné, že po jeho přečtení sidodáte odvahu k rozšíření funkcí serveru, anebo dokonce začnete pomocí vývojových nástrojů C,C++ a Java psát své vlastní klienty.

Co je LDAP?

LDAP je zkratkou Lightweight Directory Access Protocol. Jak už název napovídá, jde o „lehký" pro-tokol typu klient/server pro přístup k adresářovým službám, zejména pak ke službám založeným na X.500. LDAP běží nad TCP/IP nebo jinými přenosovými službami zaměřenými na spojení. Defi-nici LDAP naleznete v RFC2251.

Adresář se podobá databázi, avšak obsahuje více popisných informací vycházejících z „vlastností". Adresářové údaje se mnohem častěji čtou než zapisují. Jsou uspořádány tak, aby na požadavek vyhledávání ve velkém objemu dat bylo možno poskytnout rychlou odpověď. Z důvodu dostupnosti, spolehlivosti a zkrácení doby odezvy mohou být údaje v adresáři ve značném měřít-ku uloženy na více místech. Než je provedena jejich synchronizace, může docházet i k dočasným chybám v konzistenci.

Existuje mnoho metod, jak zajišťovat adresářové služby. Různé druhy informací mohou být uklá-dány v adresářích různými způsoby, vznikají různé požadavky na odkazy, dotazování a aktuali-zaci, ochranu před neoprávněným přístupem atd. Některé adresářové služby jsou lokální v ome-zeném rozsahu (např. služba finger na samostatném počítači), jiné mohou být globální a jsou poskytovány v daleko širším kontextu.

Jak pracuje?

Adresářová služba LDAP je založena na modelu klient/server. Jeden nebo několik serverů LDAP obsahují data, která vytvářejí adresářový strom neboli vnitřní strukturu („backend") databáze. Kli-ent LDAP se připojí k serveru LDAP a položí dotaz. Server buď poskytne odpověď nebo adresu, na níž klient nalezne požadovanou informaci (obvykle je to jiný server LDAP). Nezáleží na tom, ke kterému serveru LDAP se klient připojí. Pohled na adresář je vždy stejný; jméno zadané jed-nomu serveru LDAP odkazuje na stejnou položku, jako kdyby bylo zadané jinému serveru. To je důležitou vlastností globální adresářové služby, jakou je LDAP.

Vnitřní struktura, objekty a atributy LDAP

Serverový démon LDAP se jmenuje *Slapd*. Podporuje různá úložiště dat – *databáze*, které může-
te používat.Především je to BDB, což je vysoce výkonná databáze s podporou transakcí; dále LDBM – data-báze lehčího typu; SHELL je strukturované rozhraní k shellovým skriptům a PASSWD, která tvořírozhraní pro přístup k záznamům v souboru

passwd(5).
BDB využívá Sleepycat (*http://www.sleepycat.com/*), Berkeley DB 4. LDBM využívá buď Berkeley DB (*http://www.sleepycat.com/*) nebo GDBM (*http://www.gnu.org/software/gdbm/*).Transakční struktura BDB je určena pro víceuživatelský přístup k databázi s možností čtení i zápi-su a libovolné kombinace obou operací. Používá se v aplikacích, které vyžadují:

Nepřerušitelné transakce včetně hromadných změn a možnost jejich vracení do původního stavu.
Možnost obnovy při havárii systému nebo hardwarových chybách bez ztráty transakce.

V tomto dokumentu budeme pracovat s *databází BDB*. Pro import a export adresářových dat mezi servery LDAP a k popisu změn v adresáři obvykle pou-žíváme formát souboru označovaný jako LDIF, což je zkratka, která vznikla z LDAP Data Inter-change Format. Soubor LDIF ukládá data do objektově orientovaných položkových struktur. Softwarový balík LDAP obsahuje nástroj, kterým provádíme konverzi souborů LDIF do formátu BDB. Běžný soubor LDIF vypadá takto:

dn: o=TUDelft, c=NL
o: TUDelft objectclass: organization dn: cn=Luiz Malere, o=TUDelft, c=NL cn: Luiz Malere sn: Malere mail: malere@yahoo.com objectclass: person
Jak vidíte, každá položka je jednoznačně identifikovaná rozlišitelným jménem, resp. DN (*distin-guished name*). DN se skládá ze jména položky a cesty tvořené jmény, která směřují zpět ke koře-ni adresářové struktury (jako u stromu).

Třída objektů v LDAP definuje množinu *atributů*, které lze použít k definici položky. Existují tyto základní typy tříd objektů:

Skupiny v adresáři včetně neuspořádaného seznamu jednotlivých objektů nebo skupin objektů.
Umístění, např. jméno státu a popis.
Organizace v adresáři.
Lidé v adresáři.

Položka může patřit více než jedné třídě objektů. Například položka pro osobu je definovaná tří-dou objektů *person*, ale může být definovaná také atributy v třídě objektů *inetOrgPerson*, *group-OfNames* nebo *organization*. Struktura tříd objektů na serveru (což je schéma) určuje celkovýseznam požadovaných a povolených atributů pro určitou položku.

Data v adresáři představují dvojice atribut – hodnota. Každá informace je spojena s popisným atri

butem. Například atribut commonName (resp. cn) se používá k uložení jména osoby. Osoba jménemJonas Salk bude v adresáři uvedena jako:

cn: Jonas Salk

Každá osoba v adresáři je definovaná jako množina atributů v třídě objektů *person*. Jiné atributy, jimiž lze definovat tuto položku, mohou obsahovat:

givenname: Jonas surname: Salk mail: jonass@airius.com

Požadované atributy jsou atributy, které musí být uvedeny v položkách dané třídy objektů. Všech ny položky vyžadují atribut objectClass, který je seznamem tříd objektů, k němž položka patří. Povolené atributy jsou atributy, které mohou být uvedeny v položkách dané třídy objektů. Napří-klad v třídě objektů person jsou požadované atributy cn a sn. Atributy telephoneNumber, seeAl-so a userpassword jsou povolené, nikoli však požadované.

Všechny atributy mají syntaktickou definici, která popisuje typ informace poskytované daným atributem, například:

bin binární;
ces case exact string (při porovnávání řetězců musí být ve shodě i malá a velká písmena);
cis case ignore string (při porovnávání se neberou v úvahu malá/velká písmena);
tel telephone number string (jako cis, avšak navíc se ignorují mezery a pomlčky „–“);
dn jména se musí lišit.

> Poznámka
>
> Definice tříd objektů a atributů jsou v instalaci OpenLDAP obvykle v souborech schema v podadresáři schema/.

Nové verze dokumentu

Tento dokument může být opravován a aktualizován na základě ohlasů čtenářů. Nové verze toho-to návodu naleznete na: *http://www.tldp.org/HOWTO/LDAP-HOWTO.html.*

Názory a návrhy

Pokud byste měli o údajích uvedených v tomto dokumentu jakékoli pochybnosti, kontaktujte měna e-mailové adrese: <*malere@yahoo.com*>.Seznamte mne, prosím, i s případnými komentáři a návrhy...

Instalace serveru LDAP

Server nainstalujete pomocí následujících pěti kroků:

Nainstalujte požadované balíky (pokud už nejsou nainstalovány).
Stáhněte si server.
Software rozbalte.
Zkonfigurujte soubory Makefiles.
Vygenerujte server.

Vytvoření prostředí

Z důvodu plné shody s LDAPv3 vyžadují klienti i servery instalaci některých dalších balíků. Pro psaní tohoto dokumentu jsem použil systém Mandrake 9.0 s jádrem 2.4.20, ručně nainstalovaný balík Berkeley BDB a knihovny SASL.

TLS knihovny OpenSSL

TLS knihovny OpenSSL jsou běžnou součástí základního systému anebo tvoří jeho volitelnou sou-část. Oficiální url OpenSSL je: *http://www.openssl.org.*

Autentizační služby Kerberos

Klienti i servery OpenLDAP podporují autentizační služby založené na systému Kerberos. Kon-krétně OpenLDAP podporuje autentizační mechanismus SASL/GSSAPI prostřednictvím balíků Heimdal nebo MIT Kerberos V. Pokud chcete používat autentizaci SASL/GSSAPI založenou na systému Kerberos, musíte nainstalovat buď Heimdal nebo MIT Kerberos V. Heimdal Kerberos naleznete na *http://www.pdc.kth.se/heimdal*, MIT Kerberos na *http://web.mit.edu/kerberos/www.*

Používání silné autentizační služby typu Kerberos lze vřele doporučit.

Knihovny Cyrus SASL (Simple Authentication and Security Layer)

Knihovny Cyrus SASL jsou obvykle součástí základního systému nebo jsou jeho volitelnou kom-ponentou. Naleznete jej na adrese *http://asg.web.cmu.edu/sasl/sasl-library.html.* Při instalaci vyu-žívají knihovny OpenSSL a Kerberos/GSSAPI. V době psaní tohoto dokumentu byl k dispozici Cyrus SASL ve verzi 2.1.17.

Databázový software

Primární databázová struktura BDB démona slapd vyžaduje Sleepycat Software Berkeley DB (*http://www.sleepycat.com/*), verzi 4. Při konfiguraci musí být k dispozici, jinak slapd s primární databázovou strukturou nelze vygenerovat.
Je pravděpodobné, že součástí vašeho základního operačního systému nebo jeho volitelnou kom-ponentou je Berkeley DB, verze 4. Pokud není, několik verzí této databáze naleznete na stránkách Sleepycat (*http://www.sleepycat.com/download.html*). Její nejnovější verze nese v době psaní toho-to dokumentu označení 4.2.52 a lze ji doporučit. Struktura LDBM démona slapd podporuje celou řadu správců databází, např. Berkeley DB (version 3) nebo GDBM. GDBM je možno stáhnout z ftp stránek FSF na adrese ftp://ftp.gnu.org/pub/gnu/gdbm/.

Vlákna

Je téměř jisté, že váš Linux obsahuje také podporu vláken. Tuto výhodu využívá OpenLDAP. Pod-poruje posixový pthreads, Mach CThreads a řadu dalších. Pokud skript *configure* nenalezne vhod-ný subsystém vláken, ohlásí to. Kdyby tato situace nastala, nahlédněte, prosím, do části Software
– Installation – Platform Hints v často kladených otázkách (FAQ) systému OpenLDAP na adrese *http://www.openldap.org/faq/.*

TCP Wrappers

Jsou-li předinstalovány TCP wrappery (filtry pro řízení přístupu na úrovni IP), *slapd* je podporuje. Použití těchto nebo podobných přístupových filtrů na úrovni IP (např. takových, které obsahují IP firewally) lze doporučit zejména na serverech, na nichž jsou uloženy neveřejné informace.

Stažení balíku

Existují dva volně šiřitelné servery LDAP: Je to jednak server LDAP z University of Michigan a jed-nak server OpenLDAP. Také existuje Netscape Directory Server, který je zdarma pouze za určitých podmínek (například pro vzdělávací instituce). Server OpenLDAP vychází z nejnovější verze ser-veru University of Michigan a z tohoto serveru má k dispozici rozesílací seznamy a další doku-mentaci. V tomto dokumentu budeme předpokládat, že používáte server OpenLDAP.

Nejnovější verze OpenLDAP v komprimované (.zip) a archivované (.tar) podobě je k dispozici na adrese: *http://www.openldap.org*.Nejnovější verze serveru University of Michigan naleznete na adrese:ftp://terminator.rs.itd.umich.edu/ldap.
Při psaní tohoto dokumentu jsem používal verzi OpenLDAP 2.2.5. Mám operační systém Mandra

ke Linux 9.0 s jádrem 2.4.20.Na stránkách OpenLDAP také naleznete informace o vývoji tohoto produktu a stabilní verze serveru LDAP. V době aktualizace tohoto dokumentu byla poslední stabilní verzí openldap-stable-20031217.tgz (verze 2.1.25) a poslední vývojovou verzí openldap-2.2.5.tgz.

Rozbalení

Teď, když máte na počítači komprimovaný balík tar, jej můžete rozbalit. Nejprve balík zkopírujte do vhodného adresáře, například /usr/local. Dále zadejte příkaz:

```
tar xvzf openldap-2.2.5.tgz
```

anebo:

```
gunzip openldap-2.2.5.tgz | tar xvf -
```

Konfigurace

Zdrojové soubory serveru OpenLDAP jsou distribuovány s konfiguračním skriptem pro nastavení různých parametrů, např. instalačních adresářů a příznaků pro překladač a sestavovací program. V adresáři, kde máte rozbalený software, zadejte příkaz:

```
./configure --help
```

Vypíší se všechny volby, které je možno zadat skriptu configure před zahájením generování. Pro nastavení instalačních adresářů jsou důležité parametry --prefix=pref, --exec-prefix=eprefix a --bindir=dir. Když spustíte skript configure bez parametrů, obvykle provede příslušná nasta-vení sám a připraví generování do implicitního adresáře. Takže pouze zadáte:

```
./configure
```

a pozorujete, zda vše běží, jak má.

> Tip Někdy potřebujete předat skriptu configure určité parametry, např. --with-tls (aktivace používání bezpečného kanálu démonem slapd: LDAPS://). Může se stát, že máte knihov-ny SSL/TLS uložené v nestandardních adresářích. V takovém případě upozorníte skript configure na změnu umístění knihoven příkazem env. Příklad: Předpokládejte, že máte balík openssl nainstalovaný v /usr/local/openssl. Démona slapd s podporou SSL/TLS vygenerujete tímto příkazem:
>
> ```
> env CPPFLAGS=-I/usr/local/openssl/include \ LDFLAGS=-L/usr/local/openssl/lib \ configure --with-tls ...
> ```

Než spustíte skript configure, můžete příkazem env specifikovat následující vnější proměnné:

CC: alternativní překladač C.
CFLAGS: přídavné příznaky překladače.
CPPFLAGS: příznaky preprocesoru C.
LDFLAGS: příznaky sestavovacího programu.
LIBS: přídavné knihovny.

Kompilace serveru

Poté co software zkonfigurujete, můžete začít s kompilací. Nejdříve vygenerujete závislosti pomocí příkazu:

```
make depend
```

a pak už můžete zkompilovat server příkazem:

```
make
```

Běží-li vše, jak má, server bude zkompilován dle konfigurace. Pokud nikoli, vraťte se k předcho-zímu kroku a zkontrolujte konfiguraci. Také byste si měli v tom adresáři, kde jste tento software rozbalili, přečíst soubory INSTALL a README. Také zkontrolujte pokyny, které se vztahují ke skrip-tu configure – jsou umístěny na cestě doc/install/configure pod adresářem, do nějž jste roz-balili daný software.

O tom, zda je server vygenerován správně, se můžete přesvědčit pomocí sady testů (což zabere nanejvýš pár minut):

```
make test
```

Spustí se pouze testy odpovídající dané konfiguraci a ty by měly projít. Některé testy mohou být

vynechány, například test replikace.
Nyní nainstalujte binární soubory a manuálové stránky. Je možné, že to budete muset provést jako

superuživatel (záleží na tom, kam budete instalovat):

```
su root -c 'make install'
```

To je vše, vytvořili jste binární soubory serveru a několika dalších nástrojů. Konfiguraci serveru LDAP naleznete v následující kapitole.

Konfigurace serveru LDAP

Jakmile máte software nainstalovaný, můžete začít s konfigurací pro danou síť. Pro konfiguraci démona slapd v době běhu je určen soubor slapd.conf nainstalovaný v adresáři zadaném v kon-figuračním skriptu nebo implicitně v adresáři /usr/local/etc/openldap.

V této části podrobně probereme nejdůležitější konfigurační příkazy v souboru *slapd.conf*. Jejich úplný seznam naleznete v manuálových stránkách man 5 slapd.conf. Konfigurační příkazy jsou rozděleny dle toho, k čemu jsou určeny, na *globální*, *strukturové* a *databázové*. Zde naleznete popis těchto příkazů, implicitní hodnoty (pokud existují) a příklady použití.

Tvar konfiguračního souboru

Soubor slapd.conf obsahuje tři typy konfiguračních informací: globální, strukturové a databázo-vé. Nejdříve jsou uvedeny globální informace, dále následují informace, které se týkají typu struk-tury, a nakonec informace o konkrétní použité databázi.

Globální příkazy mohou být přepsány v příkazech pro strukturu a/nebo databázi, příkazy pro

strukturu mohou být přepsány databázovými příkazy.Prázdné řádky a komentáře začínající znakem # jsou ignorovány. Začíná-li řádek mezerou, jepovažován za pokračování předchozího řádku. Obecný tvar souboru slapd.conf je takovýto:

```
# globální konfigurační příkazy
<globální konfigurační příkazy>


# definice struktury
backend <typeA>
<příkazy pro strukturu>


# definice první databáze & konfigurační příkazy
database <typeA>
<příkazy pro databázi>


# definice druhé databáze & konfigurační příkazy
database <typeB>
<příkazy pro databázi>


# druhá definice databáze "typeA" & konfigurační příkazy
database <typeA>
<příkazy pro databázi>


# následují definice struktury & databáze & konfigurační příkazy
...
```

Konfigurační příkaz může mít parametry oddělené mezerami. Obsahuje-li parametr mezeru, je nutno jej uvést v uvozovkách, „takto“. Obsahuje-li parametr znak dvojitou uvozovku nebo obrácené lomítko „\“, musí před ním být obrácené lomítko „\“.

Distribuce obsahuje příklad konfiguračního souboru, který bude nainstalován v adresáři /usr/local/etc/openldap. Soubory obsahující definice schémat (typy atributů a třídy objektů) jsou také uloženy v adresáři /usr/local/etc/openldap/schema.

Globální příkazy

Příkazy popsané v této části se vztahují ke všem strukturám a databázím, ledaže by byly speciál-ně přepsány v definicích struktur a databází. Parametry, které mají být nahrazeny textem, jsou uve-deny v lomených závorkách, <>.

```
access to <what> [ by <who> <accesslevel> <control> ]+
```

Tento příkaz opravňuje k přístupu (blíže určeném v <accesslevel>) k množině položek a/nebo atributů (blíže určených ve <what>) jednomu nebo více žadatelům (blíže určeným ve <who>). Pří-klady s dalšími podrobnostmi viz kapitola „Příkazy na řízení“.

Důležité: Nejsou-li uvedeny příkazy pro oprávnění k přístupu, implicitní nastavení oprávnění k přístupu (access to * by * read) znamená, že oprávnění ke čtení mají jak autentizovaní, tak i ano-nymní uživatelé.

attributetype <RFC2252 Attribute Type Description>

Tento příkaz definuje typy atributů. Další podrobnosti naleznete na stránkách Schema Specificati-on na adrese *http://www.openldap.org/doc/admin22/schema.html.*

idletimeout <integer>

Počet vteřin čekání před vynuceným uzavřením nepoužívaného klientského spojení. Implicitní hodnota je 0 a znamená zablokování této funkce.

include <filename>

Tento příkaz zadává, že slapd má číst další konfigurační údaje z uvedeného souboru, než bude pokračovat na dalším řádku běžného souboru. Soubor uvedený v příkazu include jako parametr musí mít tvar běžného konfiguračního souboru. Obvykle se používá pro specifikaci schématu.

> Poznámka
>
> Při používání tohoto příkazu musíte být opatrní. Hloubka vnoření příkazů include není omezena a neprovádí se detekce zacyklení.

loglevel <integer>

Tento příkaz udává úroveň zaznamenávání ladicích příkazů a operační statistiky do logu (běžně připojeného ke kategorii LOCAL4 démona syslogd(8)). Aby to fungovalo, OpenLDAP musí být zkonfigurován s volbami --enable-debug (implicitně) – kromě úrovní dvou statistik, které jsou aktivovány vždy. Úrovně zápisu do logu lze zvyšovat. Vztah mezi zadanými čísly a úrovněmi ladění zjistíte buď spuštěním slapd -? nebo z následující tabulky. Možné hodnoty parametru <integer> jsou:
Úrovně ladění

Úroveň Popis

-1 aktivace veškerého ladění 0 žádné ladění 1 funkce sledování 2 zpracování ladicích paketů 4 rozsáhlé sledování 8 správa spojení 16 výpis odeslaných a obdržených paketů 32 hledání filtrů 64 konfigurační soubor 128 seznam přístupů 256 statistiky spojení/operací/výsledků 512 statistika logovacích položek 1024 tisková komunikace se strukturou shellu 2048 ladění lexikální analýzy tiskových položek

Příklad: loglevel 255, resp. loglevel -1 Do logu se bude zapisovat velké množství ladicích údajů. Implicitně: loglevel 256

objectclass <RFC2252 Object Class Description>

Tento příkaz definuje třídu objektů. Další podrobnosti naleznete na stránkách Schema Specificati-on (*http://www.openldap.org/doc/admin22/schema.html*).

referral <URI>

Tento příkaz specifikuje doporučení, které má vrátit, když slapd nenajde lokální databázi, aby mohl zpracovat požadavek.

Příklad: referral ldap://root.openldap.org

Tento příkaz odkáže nelokální dotazy na globální kořenový server LDAP projektu OpenLDAP. Důvtipní klienti mohou zadat dotaz znovu tomuto serveru, avšak většina klientů pouze chce vědět, jak má zpracovat jednoduché URL, které se skládá z názvu stanice a volitelně i rozlišitelné jmenné části.

sizelimit <integer>

Tento příkaz udává maximální počet položek, které má vrátit vyhledávací operace.Implicitně: sizelimit 500timelimit <integer>

Tento příkaz udává maximální počet vteřin (v reálném čase), jak dlouho může slapd trvat odpo-věď na požadavek vyhledávání. Není-li požadavek v tomto čase proveden, je vrácen výsledeks indikací překročení časového limitu.

Implicitně: timelimit 3600

Obecné příkazy pro strukturu

Příkazy v této kapitole se vztahují pouze na strukturu, v níž jsou definované. Jsou podporovány všemi typy struktur. Příkazy pro strukturu platí pro všechny databáze téhož typu a v závislosti na konkrétním příkazu mohou být přepsány databázovými příkazy.

backend <type>

Tento příkaz označuje začátek definice struktury; <type> by měl být některý bdb nebo jeden z jiných podporovaných typů struktur dle následujícího seznamu:

Struktury databází

Typ	Popis
bdb	transakční struktura Berkeley DB
dnssrv	struktura DNS SRV
ldbm	struktura Lightweight DBM
ldap	struktura Lightweight Directory Access Protocol (Proxy)
meta	struktura Meta Directory
monitor	struktura Monitor
passwd	zajišuje pouze čtení z passwd(5)
perl	programátorská struktura Perlu
shell	struktura Shellu (externí program)
sql	programovatelná struktura SQL

Příklad: backend bdbTento příkaz označuje začátek definice nové struktury BDB.

Obecné databázové příkazy

Příkazy v této kapitole se vztahují pouze na databázi, v níž jsou definované. Jsou podporovány všemi typy databází.

database <type>

Tento příkaz označí začátek definice nového databázového procesu; <type> by měl být jeden z typů struktur uvedených v předchozí položce.

Příklad: database bdb

Tento příkaz označuje začátek definice nového výskytu databáze se strukturou BDB.

readonly { on | off }

Tento příkaz převede databázi do režimu „read-only“, v němž lze z databáze pouze číst. Pokusy o modifikaci databáze budou odmítnuty s hlášením chyby „unwilling to perform“.

Implicitně: readonly off replica uri=ldap[s]://<hostname>[:<port>] | host=<hostname>[:<port>] [bindmethod={simple|kerberos|sasl}] ["binddn=<DN>"] [saslmech=<mech>] [authcid=<identity>]
[authzid=<identity>]
[credentials=<password>]
[srvtab=<filename>]

Tento příkaz zadává umístění kopie dané databáze. Parametr uri= zadává schéma, počítač a voli-telně i port, kde lze nalézt dceřiný proces slapd. Na místě <hostname> může být jméno domény nebo IP adresa. Není-li zadán <port>, použije se standardní číslo portu LDAP (389 nebo 636).

Poznámka

Parametr uri má přednost před host.

Parametr uri umožňuje zadat server LDAP pro uložení kopie jako LDAP URI, např. ldap://slave.example.com:389 nebo ldaps://slave.example.com:636. Parametr binddn= svazuje DN pro aktualizaci s dceřiným procesem slapd. DN by měl mít opráv-nění ke čtení i zápisu do databáze dceřiného procesu slapd. Musí také být ve shodě s příkazem updatedn v konfiguračním souboru démona slapd. Obecně by tento DN *neměl být* totožný s rootdn v hlavní databázi. Vzhledem k tomu, že DN pravděpodobně obsahují mezery, celý řetě-zec „binddn=<DN>“ by měl být uzavřen ve dvojitých uvozovkách.

Parametr bindmethod je simple nebo kerberos nebo sasl v závislosti na tom, zda při připojování k dceřinému procesu slapd je

použita jednoduchá autentizace na základě hesla nebo Kerberos nebo SASL.

Jednoduchá autentizace by se neměla používat, pokud není v činnosti příslušná ochrana integrity a soukromí (např. TLS nebo IPSEC). Při jednoduché autentizaci musí být zadány binddn a para-metry s autentizačními údaji.

Autentizační mechanismus SASL má přednost před autentizací Kerberos, konkrétně jde o mecha-nismy KERBEROS_V4 a GSSAPI. Autentizace pomocí mechanismu Kerberos vyžaduje parametry binddn a srvtab.

Obecně lze doporučit autentizaci SASL. Pokud ji použijete, mechanismus musíte zadat v paramet-ru saslmech. V závislosti na vybraném mechanismu lze pomocí authcid, resp. credentials, zadat autentizační totožnost a/nebo autentizační údaje. K zadání autorizační totožnosti slouží parametr authzid.

Další podrobnosti o tomto příkazu naleznete na stránkách Replication with Slurpd (*http://www.openldap.org/doc/admin22/replication.html*).

replogfile <filename>

Tento příkaz zadává jméno replikačního logovacího souboru, do nějž bude slapd zapisovat změny. Do replikačního logu obvykle zapisuje slapd a čte z něho slurpd. Tento příkaz se použí-vá zejména tehdy, když probíhá replikace databáze pomocí slurpd. Můžete jej však použít i k vytvoření transakčního logu, když slurpd není v činnosti. V takovém případě budete ovšem muset ze souboru pravidelně odstraňovat nepotřebná data, jinak nekontrolovaně poroste.

Další podrobnosti o tomto příkazu naleznete na stránkách Replication with Slurpd (*http://www.openldap.org/doc/admin22/replication.html*).

rootdn <dn>

Tento příkaz zadává DN, které nepodléhá žádným kontrolám přístupu k databázi ani jiným admi-nistrativním omezením. Musí mít přístup ke všem položkám v adresáři. Může se odkazovat na totožnost SASL.

Příklad (položky): rootdn “cn=Manager, dc=example, dc=com” Příklad (SASL): rootdn “uid=root,cn=example.com,cn=digest-md5,cn=auth”

rootpw <password>

Tento příkaz je možno použít k zadání hesla pro rootdn (když je rootdn nastaveno na DN uvnitř databáze).
Příklad: rootpw secret Je také přípustné provést transformaci hesla do tvaru RFC 2307. Transformaci může provést napří-klad slappasswd.

Příklad: rootpw {SSHA}ZKKuqbEKJfKSXhUbHG3fG8MDn9j1v4QN Transformace byla

provedena příkazem slappasswd -s secret.

suffix <dn suffix>

Tento příkaz zadává příponu DN dotazů, které budou předávány struktuře této databáze. Můžebýt zadáno více řádků s příponami, pro každou definici databáze musí být zadán alespoň jeden.Příklad: suffix “dc=example, dc=com”Struktuře budou předány dotazy se zakončením DN „dc=example, dc=com“.

> Poznámka
>
> Když je vybrána struktura pro předání dotazu, slapd se ve všech definicích databází podí-vá na řádky s příponami v tom pořadí, v jakém se nacházejí v souboru. Je-li tedy jedna pří-pona databáze předponou jiné, musí být v konfiguračním souboru uvedena za ní.

syncrepl

Tento příkaz se používá k synchronizaci replikované databáze s hlavní databází, takže obsah re

plikované databáze je udržován v aktuálním stavu v souladu s hlavní databází.Tento příkaz není v tomto dokumentu popsán podrobně, protože konfigurujeme samostatný ser-ver LDAP. Další informace o tomto příkazu naleznete na stránkách LDAP Sync Replication(*http://www.openldap.org/doc/admin22/syncrepl.html*).

updatedn <dn>

Tento příklad se vztahuje pouze na dceřiný proces slapd. Určuje přípustná DN, jež mohou pro-vádět změny v kopii databáze. Tak může být DN slurpd připojeno při provádění změn ke kopiinebo k DN spojenému s totožností SASL.

Příklad (položky): updatedn “cn=Update Daemon, dc=example, dc=com”Příklad (SASL): updatedn “uid=slurpd,cn=example.com,cn=digest-md5,cn=auth”Další podrobnosti o tomto příkazu naleznete na stránkách Replication with Slurpd (*http://www.openldap.org/doc/admin22/replication.html*).

updateref <URL>

Tento příkaz je použitelný pouze v dceřiném procesu slapd. Zadává URL, jež má být vráceno kli-entům, kteří zaslali požadavek na aktualizaci kopie databáze. Je-li uvedeno několik URL, předává je všechna.

Příklad: updateref ldap://master.example.net

Příkazy databáze BDB

Tato kategorie příkazů se vztahuje pouze na databázi BDB. Musí tedy být za řádkem „database bdb“ a před řádky „backend“ a „database“, které následují. Úplný přehled konfiguračních příkazů BDB naleznete na manuálových stránkách slapd-bdb (man slapd-bdb).

directory <directory>

Tento příkaz zadává adresář, v němž jsou umístěny soubory s databází BDB a příslušné indexy. Implicitně: directory

/usr/local/var/openldap-data

sessionlog <sid> <limit>

Tento příkaz definuje úložiště logů daného sezení na serveru poskytovatele kopie syncrepl, které obsahuje informace o sledovaných položkách v kopii databáze určených pomocí <sid>. První požadavek na hledání syncrepl má v cookie, které vytvořilo úložiště logu daného sezení na ser-veru poskytovatele, stejnou hodnotu <sid>. Počet položek v úložišti je omezen parametrem <limit>. Přebytečné položky jsou z úložiště odstraňovány v pořadí FIFO. <sid> a <limit> jsou nezá-porná celá čísla. <sid> může mít nejvýš tři dekadické číslice.

Synchronizace obsahu provedená v předchozím sezení mohla z důvodu omezení synchronizačních přenosů použít úložiště logů. Není-li kopie databáze tak stará, aby ji nebylo možné aktualizovat z dat uložených v úložišti sezení, poskytovatel slapd pošle zákazníkovi slapd jména sledovaných položek společně s položkami, jež byly přidány nebo modifikovány v kopii databáze. Je-li však databáze natolik zastaralá, že ji nelze z tohoto úložiště aktualizovat, poskytovatel slapd pošle jména sledovaných položek, které nebyly změněny, společně s těmi, které byly změněny. Zákazník pak v kopii odstraní ty položky, které nejsou v obsahu poskytovatele označeny jako přítomné.

Další informace o tomto příkazu naleznete na stránkách LDAP Sync Replication (*http://www.openldap.org/doc/admin22/syncrepl.html*).

Databázové příkazy LDBM

Příkazy v této kategorii se vztahují pouze k struktuře databáze LDBM. Musí tedy být až za řádkem „database ldbm“ a před řádky „database“ a „backend“, které následují. Úplný přehled konfigurač-ních příkazů LDBM naleznete na manuálových stránkách slapd-ldbm (man slapd-ldbm).

cachesize <integer>

Tento příkaz zadává velikost rychlé vyrovnávací paměti (cache) v položkách používané procesem,který byl vytvořen strukturou databáze LDBM.Implicitně: cachesize 1000

dbcachesize <integer>

Tento příkaz zadává velikost rychlé vyrovnávací paměti (cache) v bajtech přiřazené indexovému souboru. Pokud příslušná databázová metoda tuto paměť nepotřebuje, příkaz je bez komentáře ignorován. Zvětšíte-li hodnotu parametru, zvýší se spotřeba paměti, avšak také může dramaticky vzrůst výkon, a to zvlášť při modifikaci databáze nebo při vytváření indexů.

Implicitně: dbcachesize 100000

dbnolocking

Je-li tato volba přítomna, blokuje zamykání databáze. Její aktivace může zvýšit výkon za cenu snížení bezpečnosti databáze.

dbnosync

Tato volba ruší okamžitou synchronizaci disku s operační pamětí při změně jejího obsahu. Aktivace této volby může zvýšit výkon za cenu snížení bezpečnosti databáze.

directory <directory>

Tento příkaz zadává adresář se soubory LDBM obsahujícími databázi a příslušné indexové sou-bory. Implicitně:

directory /usr/local/var/openldap-data

index {<attrlist> | default} [pres,eq,approx,sub,none]

Tento příkaz zadává indexy, které mají být udržovány pro daný atribut. Je-li zadáno pouze <attr

list>, jsou udržovány implicitní indexy.

Příklad:

index default pres,eq index uid index cn,sn pres,eq,sub index objectClass eq

První řádek nastaví implicitní množinu indexů tak, aby byly udržovány pro present a equality. Druhý řádek říká, že implicitní množina indexů (pres,eq) má být udržována pro typ atributu uid. Třetí řádek říká, že indexy present, equality a substring mají být udržovány pro typy atributů cn a sn. Čtvrtý řádek říká, že index equality má být udržován pro typ atributu objectClass. Implicitně nejsou udržovány žádné indexy. Obecně lze doporučit, aby byl udržován alespoň index equality pro objectClass. index objectClass eq

mode <integer>

Tento příkaz zadává režim ochrany nově vytvořených indexových souborů. Implicitně: mode 0600

Příklady na řízení přístupu

Příkaz *access* je velice silným nástrojem pro řízení přístupu. V této kapitole si ukážeme několik příkladů použití. Nejdříve několik jednodušších příkladů:

access to * by * read

Tento příkaz zadává, že všichni mohou číst.
Následující příklad znázorňuje, jak se dvěma příkazy access s regulárními výrazy vyberou položky pomocí DN, přičemž je důležité pořadí těchto příkazů.

```
access to dn=".*, o=U of M, c=US"
by * search
access to dn=".*, c=US"
by * read
```

Položky v podstromě pod c=US mohou číst s výjimkou položek v podstromě pod „o=U z M, c=US“, které mohou vyhledávat. c=US nemá povolen žádný přístup, neboť žádný příkaz access není shodný s tímto DN. Pokud byste tyto příkazy provedli v opačném pořadí, příkaz s U-M by nikdy nebyl ve shodě, neboť všechny položky U-M jsou současně položkami c=US.
Stejný přístup lze zadat i takto:

```
access to dn.children="dc=example,dc=com"
        by * search
        access to dn.children="dc=com"
        by * read
```

Položky v podstromě pod dc=com mohou číst s výjimkou položek v podstromě pod dc=exam-ple,dc=com, které mohou vyhledávat. dc=com nemá povolen žádný přístup, neboť žádný příkaz access není shodný s tímto DN. Pokud byste tyto příkazy provedli v opačném pořadí, druhý pří-kaz by se nikdy neprovedl, neboť všechny položky pod dc=example,dc=com jsou současně pod položkami dc=com.

> Poznámka
>
> Všimněte si také, že pokud se příkaz neprovede nebo klauzule „by <who>“ není ve shodě, přístup není povolen. To znamená, že všechny příkazy access to jsou implicitně ukončeny klauzulí * none a všechny seznamy přístupů jsou implicitně ukončeny příkazem access to * by * none.

V následujícím příkladu si opět ukážeme, jak důležité je pořadí příkazů access a současně i klau-zulí „by <who>“. Příkaz také znázorňuje, jak můžete pomocí selektoru atributů poskytnout přístup k určitému atributu a k různým selektorům <who>.

```
access to dn.subtree="dc=example,dc=com" attr=homePhone
        by self write
        by dn.children=dc=example,dc=com" search
        by peername=IP:10\..+ read access to dn.subtree="dc=example,dc=com"
        by self write
        by dn.children="dc=example,dc=com" search
        by anonymous auth
```

Tento příklad se vztahuje k položkám v podstromě „dc=example,dc=com“. Tyto položky mohou zapisovat do svých atributů kromě homePhone, položky pod položkami example.com se mohou prohledávat. Nikdo jiný nemá přístup (implicitně platí * none) kromě autentizace a autorizace, jež jsou vždy prováděny anonymně. Položka může zapisovat do atributu homePhone, mohou jej pro-hledávat položky pod example.com, mohou jej číst klienti připojení ze sítě 10 a jinak jej nemůže číst nikdo (implicitně platí * none). Všechny ostatní přístupy jsou implicitně odmítnuty klauzulí * by * none.

Někdy je vhodné povolit určitému DN, aby se samo smělo přidávat k atributu a odstraňovat se z něj. Když například chcete vytvořit skupinu a povolit uživatelům, aby směli k atributu member přidávat pouze svoje DN a odstraňovat je, můžete to provést například takovýmto příkazem:

```
access to attr=member,entry by dnattr=member selfwrite
```

Selektor <who> dnattr říká, že přístup platí pro položky v atributu member. Selektor zápisu sama do sebe říká, že tito členové mohou do atributu přidávat nebo z něj odstraňovat pouze svoje DN, jiné hodnoty nikoli. Přidávání atributu položky je nutné, protože přístup k položce znamená přístup ke všem jejím atributům.

Mnoho dalších informací o řízení přístupu je uvedeno v OpenLDAP Administrator's Guide. Chce-te-li se s nimi seznámit, nahlédněte na stránky Access Control (*http://www.openldap.org/doc/ admin22/slapdconfig.html#Access Control*).

Příklad konfiguračního souboru

Následuje příklad konfiguračního souboru proloženého vysvětlujícím textem. Soubor definuje dvě databáze pro zpracování různých částí stromu X.500; obě jsou procesy BDB. Čísla řádků slouží pouze odkazům v textu a ve skutečnosti nejsou součástí souboru. Nejprve tedy globální konfigurační část:

```
# example config file - global configuration section
include /usr/local/etc/schema/core.schema
referral ldap://root.openldap.org
access to * by * read
```

První řádek je komentář. Druhý řádek obsahuje jiný konfigurační soubor, v němž je definice základního schématu. Příkaz referral na třetím řádku znamená, že dotazy, které nejsou lokální vzhledem k žádné z níže definovaných databází, budou odkázány na server LDAP, který běží na standardním portu (389) na počítači root.openldap.org.

Na řádku 4 je zadán globální přístup. Vztahuje se na všechny položky (po všech příkazech pro řízení přístupu, které se vztahují k databázi). Následující část konfiguračního souboru definuje strukturu BDB, která bude zpracovávat dotazy v části stromu „dc=example,dc=com". Databáze bude replikovaná na dva dceřiné slapd, jeden na slave1, druhý na slave2. Indexy budou udržovány pro několik atributů a atribut userPassword bude chráněn proti neautorizovanému přístupu.

```
# BDB definition for the example.com
database bdb
suffix "dc=example,dc=com"
directory /usr/local/var/openldap-data
rootdn "cn=Manager,dc=example,dc=com"
rootpw secret
# replication directives
replogfile /usr/local/var/openldap/slapd.replog
replica uri=ldap://slave1.example.com:389
binddn="cn=Replicator,dc=example,dc=com"
bindmethod=simple credentials=secret
replica uri=ldaps://slave2.example.com:636
binddn="cn=Replicator,dc=example,dc=com"
bindmethod=simple credentials=secret
# indexed attribute definitions
index uid pres,eq
index cn,sn,uid pres,eq,sub
index objectClass eq
# database access control definitions
access to attr=userPassword
by self write
by anonymous auth
by dn.base="cn=Admin,dc=example,dc=com" write
by * none
access to *
by self write
by dn.base="cn=Admin,dc=example,dc=com" write
by * read
```

Řádek 5 je komentář. Začátek definice databáze je označen klíčovým slovem databáze na řádku 6. Řádek 7 specifikuje příponu DN pro dotazy předávané této databázi. Řádek 8 specifikuje adresář, v němž se budou nacházet databázové soubory.Řádky 9 a 10 identifikují databázovou položku „super user" a heslo, které k ní náleží. Tato polož-ka není předmětem řízení přístupu, omezení velikosti ani časové prodlevy. Nezapomeňte, prosím,rootpw zašifrovat pomocí slappasswd.

Příklad: rootpw {SSHA}Jq4xhhkGa7weT/0xKmaecT4HEXsdqiYA

Řádky 11 až 18 zajišťují replikace. Další informace o tomto příkazu viz odkaz na stránky Replica-tion (*http://www.openldap.org/doc/admin22/replication.html*).Řádky 20 až 22 udávají, které indexy se mají udržovat pro různé atributy.Řádky 24 až 32 specifikují řízení přístupu pro položky v této databázi. Vzhledem k tomu, že je to první databáze, řízení se také vztahuje na položky, které nejsou obsaženy v žádné databázi (např.Root DSE). Pro všechny položky platí, že do atributu userPassword může psát položka samaa položka „admin“. Lze ji používat pro účely autentizace a autorizace, jinak ji ale nelze číst. Dovšech ostatních atributů může psát položka sama a položka „admin“, avšak mohou ji číst všichniuživatelé (autentizovaní i neautentizovaní).

Následující část ukázkového konfiguračního souboru definuje jinou databázi BDB. Ta zpracovává dotazy z podstromu dc=example,dc=net, avšak je spravovaná stejnou entitou jako první databáze. Všimněte si, že bez řádku 39 by kvůli globálnímu přístupovému pravidlu na řádku 4 bylo povoleno čtení.

```
# BDB definition for example.net
database bdb
suffix "dc=example,dc=net"
directory /usr/local/var/openldap-data-net
rootdn "cn=Manager,dc=example,dc=com"
index objectClass eq
access to * by users read
```

Provozování serveru LDAP

Démon *slapd* je určen k provozování na samostatném serveru. To umožňuje tomuto serveru vyu-žít rychlou vyrovnávací paměť, paralelně spravovat problémy podpůrných databází a šetřit syste-mové prostředky. Spuštění pomocí démona (x)inetd není podporováno.

Volby příkazového řádku

Slapd podporuje řadu voleb příkazového řádku, jak je podrobně uvedeno v manuálových strán-kách. V této kapitole se budeme detailně věnovat jen několika často používaným volbám:

-f <filename>

Tato volba zadává alternativní konfigurační soubor pro slapd. Implicitní je obvykle /usr/local/etc/openldap/slapd.conf.

-h <URLs>

Tato volba zadává alternativní konfigurace naslouchání. Implicitní je ldap:///, z níž plyne LDAP přes TCP na všech rozhraních na implicitním portu LDAP 389. Můžete zadat zvláštní dvojice počí-tač – port nebo jiná schémata protokolu (např. ldaps:// nebo ldapi://). Například -h „ldaps:// ldap://127.0.0.1:667“ vytvoří dva posluchače: jednoho pro LDAP přes SSL na všech rozhraních na implicitním portu LDAP/SSL 636 a jednoho pro LDAP přes TCP na rozhraní lokálního počítače (prázdná smyčka) na portu 667. Počítače mohou být specifikovány v dekadickém tvaru IPv4 s tečkami nebo pomocí jmen počítačů. Hodnoty portů musí být číselné.

-n <service-name>

Tato volba zadává jméno služby pro logování a podobné účely. Implicitní jméno služby je slapd.

-l <syslog-local-user>

Tato volba zadává lokálního uživatele pro syslog(8). Hodnotami může být LOCAL0, LOCAL1, LOCAL2, ... a LOCAL7. Implicitní hodnotou je LOCAL4. Tuto volbu nemusí podporovat všechny systémy, podrobnosti viz kapitola „Logy“.

-u user -g group

Tyto volby zadávají uživatele a skupinu, kteří budou provozovat slapd; user může být buď jméno uživatele nebo uid; group může být jméno skupiny nebo gid.

-r directory

Tato volba zadává adresář v době běhu. Po otevření posluchačů, avšak před čtením konfigurač-ních souborů a inicializací struktury si slapd příkazem chroot vytvoří z tohoto adresáře nový koře-nový adresář.

-d <level> | ?

Tato volba nastaví úroveň ladění slapd na <level>. Je-li level znak „?“, slapd vytiskne různé režimy ladění a ukončí se bez ohledu na případné další volby. Úrovně ladění popisuje následující tabulka:

Úrovně ladění

Úroveň Popis

-1 aktivace veškerého ladění 0 žádné ladění 1 funkce sledování 2 zpracování ladicích paketů 4 rozsáhlé sledování 8 správa spojení 16 výpis odeslaných a obdržených paketů 32 hledání filtrů 64 konfigurační soubor 128 seznam přístupů 256 statistiky spojení/operací/výsledků 512 statistika logovacích položek 1024 tisková komunikace se strukturou shellu 2048 ladění lexikální analýzy tiskových položek

Můžete aktivovat několik úrovní tak, že zadáte volbu pro každou požadovanou úroveň. Anebo, vzhledem k tomu, že úroveň ladění je aditivní, můžete si ji spočítat sami. Když tedy například chcete sledovat volání funkcí a současně zpracovávaný konfigurační soubor, úroveň nastavíte na součet těchto dvou úrovní (v tomto případě -d 65). Anebo můžete přenechat tento výpočet na slapd (např. -d 1 -d 64). Další podrobnosti viz soubor ldap.h.

Poznámka

Aby bylo možno tyto úrovně zadávat, slapd musí být přeložen s volbou -DLDAP_DEBUG

Spuštění serveru LDAP

Obecně spustíte slapd takto:

/usr/local/etc/libexec/slapd [<option>]*

kde /usr/local/etc/libexec je určen příkazem configure a <option> je jedna z voleb popsa-ných shora (viz slapd(8)). Pokud úroveň ladění nezadáte (a to i úroveň 0), slapd automaticky vytvoří nový proces, odpojí se od řídicího terminálu a spustí se na pozadí.

Zastavení serveru LDAP

slapd bezpečně zastavíte například příkazem:

kill -INT `cat $(ETCDIR)/slapd.pid`

Drastičtější zastavení by mohlo databázi narušit, neboť před ukončením činnosti musí slapd vyprázdnit různé bufery. Poznamenejme, že slapd zapíše svůj pid do souboru slapd.pid, který je v adresáři uvedeném v souboru slapd.conf, například /usr/local/var/slapd.pid.

Slapd také zapíše svoje parametry do souboru slapd.args, který je v adresáři uvedeném v sou-boru slapd.conf, například /usr/local/var/slapd.args.

Vytvoření a údržba databáze

V této kapitole se dočtete, jak máte vytvořit databázi slapd od úplného počátku. Databázi může-te vytvořit dvěma způsoby. Zaprvé on-line pomocí LDAP. Při použití této metody pouze spustíte slapd a pomocí klienta LDAP, kterého si vyberete, přidáte položky. Tato metoda je vhodná pro relativně malé databáze (v závislosti na požadavcích to může být několik set nebo tisíc položek). Tato metoda funguje u těch typů databází, které podporují aktualizaci.

Druhou metodou je vytvoření databáze v off-line pomocí speciálních nástrojů obsažených ve slapd. Tato metoda je nejlepší tehdy, když máte vytvořit mnoho tisíc položek, jejichž vytváření by trvalo pomocí LDAP nepřijatelně dlouho, nebo když nutně potřebujete zcela prázdnou databázi, do níž nemůže nikdo nic zapsat při vytváření. Poznamenejme ještě, že tyto nástroje nepodporují všechny typy databází.

Vytváření databáze v době běhu (on-line)

Balík OpenLDAP obsahuje nástroj jménem ldapadd, který můžete použít ke vkládání položek v době běhu serveru LDAP. Chcete-li vytvářet databázi v režimu on-line, položky můžete přidávat tímto nástrojem (avšak nejen jím, k tomuto účelu existují i jiní klienti mimo balík OpenLDAP, například Ldap Browser, který najdete na *http://www.iit.edu/~gawojar/ldap/*). Po přidání prvních položek můžete pokračovat s přidáváním pomocí ldapadd i nadále. Před spuštěním slapd byste se měli přesvědčit, zda jste v souboru slapd.conf zadali tuto konfigurační volbu:

suffix <dn>
Jak je popsáno v kapitole „Obecné databázové příkazy“, tato volba říká, které položky má tato databáze obsahovat. Zde byste si měli nastavit DN kořene podstromu, který vytváříte. Například:

suffix “o=TUDelft, c=NL”

Měli byste se přesvědčit, zda uvádíte adresář, v němž budou vytvářeny indexové soubory:

directory /usr/local/tudelft

Adresář musí být vytvořen s příslušnými přístupovými právy – slapd musí mít oprávnění k zápisu. Slapd musí být zkonfigurován tak, že se k němu budete moci připojit jako uživatelský adresář s oprávněním přidávat položky. Můžete si zkonfigurovat adresář s podporou superuživatele nebo uživatele root jenom pro tento účel. V definici databáze můžete například zadat tyto dvě volby:

```
rootdn <dn>
rootpw <passwd> /* Nezapomeňte zde použít heslo SHA !!! */
```

Těmito volbami zadáte DN a heslo, jež použijete k autentizaci položky „superuživatel“ v databázi (tj. položka, která může dělat všechno). Zadané DN a heslo budou fungovat vždy, bez ohledu na to, zda daná položka skutečně existuje a má dané heslo. Tím je vyřešen problém slepice a vejce, neboli jak autentizovat a přidávat položky v době, kdy ještě žádné neexistují.

Slapd pochopitelně pozná, jestli používáte šifrované heslo SHA-1 nebo příkaz rootpw. Já použí-vám třídu v jazyce Java, která generuje hesla SHA-1 – je však možné generovat hesla příkazem slappasswd:

```
slappasswd -h {SHA} rootpw “{SHA}5en6G6MezRroT3XKqkdPOmY/BfQ=”
```

Například:

```
rootdn “cn=Manager,dc=example,dc=com”
rootpw “{SHA}5en6G6MezRroT3XKqkdPOmY/BfQ=”
```

Implicitním výstupem slappasswd je generování hesel Secure Hash (SSHA), v tomto případě nemusíte předávat parametr -h, stačí zavolat přímo slappasswd. Používáte-li k identifikaci proti LDAP autentizační mechanismus SASL, řádek rootpw můžete zru-šit. Další podrobnosti jsou uvedeny v kapitolách „Obecné databázové příkazy“ a „Autentizacepomocí LDAP”.

Nakonec byste se měli přesvědčit, zda definice databáze obsahuje i definici těch indexů, které potřebujete:

```
index {<attrlist> | default} [pres,eq,sub,none]
```

Abyste například mohli indexovat atributy cn, sn, uid a objectclass, je nutno zadat tyto konfigurační řádky indexů:

```
index cn,sn,uid pres,eq,sub index objectClass pres,eq
```

> Poznámka
>
> Nikoli všechny typy indexů jsou vhodné pro všechny typy atributů. Příklady naleznete v kapitole „Datové příkazy LDBM“.

Jakmile jste si vše zkonfigurovali podle svých představ, spusťte slapd, spojte se s klientem LDAP a začněte přidávat položky. Kdybyste například chtěli přidat položku TUDelft následovanou položkou Postmaster pomocí nástroje ldapadd, vytvoříte soubor /tmp/newentry, který bude obsahovat:

```
o=TUDelft, c=NL objectClass=organization description=Technical University of Delft Netherlands

cn=Postmaster, o=TUDelft, c=NL objectClass=organizationalRole cn=Postmaster description= TUDelft postmaster - postmaster@tudelft.nl
```

a položku pak vytvoříte například příkazem:

```
ldapadd -f /tmp/newentry -x -D “cn=Manager, o=TUDelft, c=NL” -w secret
```

Příkaz shora předpokládá, že jste nastavili rootdn na „cn=Manager, o=TUDelft, c=NL“ a rootpw na „secret“ (SHA-1 může být zašifrováno v slapd.conf). Nechcete-li psát heslo na příkazový řádek, místo volby -w password zadejte v příkazu ldapadd volbu -W a program si heslo vyžádá:

```
ldapadd -f /tmp/newentry -x -D “cn=Manager, o=TUDelft, c=NL” -W Enter LDAP Password:
```

Nezávislé vytváření databáze (off-line)

Druhá metoda vytváření databáze probíhá off-line prostřednictvím nástrojů databáze popsaných shora. Tato metoda je vhodná, když máte vytvářet několik tisíc položek, jejichž vytváření by shora popsanou metodou LDAP trvalo nepřiměřeně dlouho. Tyto nástroje čtou konfigurační soubor slapd a vstupní soubor LDIF, který obsahuje textovou podobu položek, jež mají být přidány. Data-bázové soubory těch databází, které podporují tyto nástroje, jsou vytvářeny přímo (jinak je nutno použít on-line metodu popsanou shora). V konfiguračním souboru je nejdříve nutno nastavit několik důležitých konfiguračních voleb:

```
suffix <dn>
```

Jak je uvedeno v předchozí kapitole, tato volba říká, které položky má tato databáze obsahovat. Zde byste si měli nastavit DN

kořene podstromu, který vytvářítе. Například:

suffix “o=TUDelft, c=NL”

Měli byste se přesvědčit, zda uvádíte adresář, v němž budou vytvářeny indexové soubory:

directory /usr/local/tudelft

Nakonec byste se měli přesvědčit, zda definice databáze obsahuje i definici těch indexů, které potřebujete.

index {<attrlist> | default } [pres,eq,approx,sub,none]

Například:

index cn,sn,uid pres,eq,sub index objectClass eq

Tím vytvoříte indexy presence, equality a substring pro atributy cn, sn a uid a index equality pro atribut objectClass. Další informace o této volbě viz kapitola popisující konfigurační soubor. Jakmile jste si vše zkonfigurovali podle svých představ, pomocí programu slapadd (man 8 sla-padd) vytvoříte primární databázi a příslušné indexy:

slapadd -l <inputfile> -f <slapdconfigfile> [-d <debuglevel>] [-n <integer>|-b <suffix>]

Parametry mají tento význam:

-l <inputfile>

Zadání vstupního souboru LDIF, který obsahuje přidávané položky v textovém tvaru (viz také další kapitola).

-f <slapdconfigfile>

Zadání konfiguračního souboru slapd, který říká, kde a jaké indexy mají být vytvořeny, atd.

-d <debuglevel>

Nastavení ladicího režimu dle <debuglevel>. Úrovně ladění jsou stejné jako v případě slapd, podrobnosti viz kapitola „Volby příkazového řádku“.

-n <databasenumber>

Volitelný parametr udávající, která databáze má být modifikována. První databáze v konfigurač-ním souboru je databáze č. 1, druhá je č. 2 atd. První databáze je implicitní. Tento parametr nelze použít současně s parametrem -b.

-b <suffix>

Volitelný parametr udávající, která databáze má být modifikována. Číslo databáze se určí porov-náváním zadané přípony s příkazem pro příponu databáze. Tento parametr nelze použít součas-ně s parametrem -n.

Někdy je nutné obnovit indexy (např. po změně slapd.conf). K tomu slouží program slapindex (man 8 slapindex), volá se takto:

slapindex -f <slapdconfigfile> [-d <debuglevel>] [-n <databasenumber>|-b <suffix>]

kde volby -f, -d, -n a -b mají stejný význam jako v programu slapadd(1). Program slapindex obnoví všechny indexy na základě stávajícího obsahu databáze.Program slapcat slouží k výpisu databáze do souboru LDIF. To může být užitečné, když chcetevytvořit zálohu, v níž je možno číst, anebo ji editovat v off-line režimu. Program spustíte takto:

slapcat -l <filename> -f <slapdconfigfile> [-d <debuglevel>] [-n <databasenumber>|-b <suffix>]

kde -n a -b se používá k výběru databáze v souboru slapd.conf pomocí parametru -f. Odpoví-dající výstup LDIF je vypsán na standardní výstup nebo do souboru zadaného volbou -l.

Více o formátu LDIF

Formát LDIF (LDAP Data Interchange Format) se používá k zápisu položek v jednoduchém textovém tvaru. Položka má základní tvar:

#comment dn: <distinguished name> <attrdesc>: <attrvalue>
<attrdesc>: <attrvalue> ...

Řádky začínající znakem # jsou poznámky. Popis atributu (attrdesc) může být jednoduchý typ atri-butu, např. cn nebo objectClass nebo 1.2.3 (OID s typem atributu), nebo může obsahovat různé volby, např. cn;lang_en_US nebo userCertificate;binary.

Řádek může pokračovat na dalším řádku mezerou nebo tabelátorem. Například:

```
dn: cn=Barbara J Jensen, dc=example, dc= com cn:
  Barbara J Jensen
```

je shodné s:

dn: cn=Barbara J Jensen, dc=example, dc=com cn: Barbara J Jensen

Různé hodnoty atributu se uvádějí na samostatných řádcích, např.:

cn: Barbara J Jensen cn: Babs Jensen

Obsahuje-li <attrvalue> netisknutelný znak nebo začíná-li mezerou, dvojtečkou („:“) nebo znakem „menší než“ („<“), za <attrdesc> následují dvě dvojtečky a hodnota zakódovaná v base64. Napří-klad hodnota „begins with a space“ bude zakódována takto:

cn:: IGJlZ2lucyB3aXRoIGEgc3BhY2U=

Také můžete uvést URL, která obsahuje hodnotu atributu. Například následující řádek znamená, že hodnota jpegPhoto je v souboru /path/to/file.jpeg.

cn:< file://path/to/file.jpeg

Položky jsou od sebe odděleny prázdnými řádky. Zde je příklad souboru LDIF se třemi položkami:

Barbara's Entry dn: cn=Barbara J Jensen, dc=example, dc=com cn: Barbara J Jensen cn: Babs Jensen objectClass: person sn: Jensen

Bjorn's Entry dn: cn=Bjorn J Jensen, dc=example, dc=com cn: Bjorn J Jensen cn: Bjorn Jensen objectClass: person sn: Jensen # Base64 encoded JPEG photo jpegPhoto:: /9j/4AAQSkZJRgABAAAAAQABAAD/2wBDABALD A4MChAODQ4SERATGCgaGBYWGDEjJR0oOjM9PDkzODdASFxOQ ERXRTc4UG1RV19iZ2hnPk1xeXBkeFxlZ2P/2wBDARESEhgVG

Jennifer's Entry dn: cn=Jennifer J Jensen, dc=example, dc=com cn: Jennifer J Jensen cn: Jennifer Jensen objectClass: person sn: Jensen # JPEG photo from file jpegPhoto:< file://path/to/file.jpeg

Všimněte si, že atribut jpegPhoto v položce Bjorn je zašifrován pomocí base64 a v položce Jenni

fer je určen URL.Pravostranné mezery se z hodnot v souboru LDIF neodstraňují a ani se žádné mezery nekompri-mují. Pokud je v datech nechcete, nedávejte je tam.

Nástroje ldapsearch, ldapdelete a ldapmodify

ldapsearch – je shellové rozhraní ke knihovnímu volání ldap_search(3). Tento nástroj je vhod-ný k hledání položek ve struktuře LDAP. Stručný výtah volání vypadá takto (význam voleb viz manuálové stránky ldapsearch):

ldapsearch [-n] [-u] [-v] [-k] [-K] [-t] [-A] [-B] [-L] [-R] [-d debuglevel] [-F sep] [-f file] [-x] [-D binddn] [-W] [-w bindpasswd] [-h ldaphost] [-p ldapport] [-b searchbase] [-s base|one|sub] [-a never|always|search|find] [-l timelimit] [-z sizelimit] filter [attrs...]

ldapsearch otevře spojení na server LDAP, naváže je a provede hledání pomocí filtru *filter*. Filtr musí odpovídat definici řetězcových filtrů LDAP v RFC 1558. Najde-li ldapsearch jednu nebo více položek, přečtou se specifikované atributy a tyto položky a hodnoty jsou vytisknuty na standard-ním výstupu. Nejsou-li uvedeny žádné attrs, všechny atributy jsou vráceny.

ldapsearch -x -b ‘o=TUDelft,c=NL’ ‘objectclass=*’

ldapsearch -b ‘o=TUDelft,c=NL’ ‘cn=Rene van Leuken’

ldasearch -u -b ‘o=TUDelft,c=NL’ ‘cn=Luiz Malere’ sn mail

Volba -b znamená bázi hledání (počáteční bod hledání), volba -u znamená uživatelsky přátelské
výstupní informace a volba -x je používána k jednoduché autentizaci.ldapdelete – je shellové rozhraní ke knihovnímu volání ldap_delete(3). Tato utilita je vhodnák rušení položek ve struktuře LDAP.
Stručný výtah volání vypadá takto (význam voleb viz manuálové stránky ldapdelete):

ldapdelete [-n] [-v] [-k] [-K] [-c] [-d debuglevel] [-f file] [-D binddn] [-W] [-w passwd] [-h ldaphost] [-p ldapport] [dn]...

ldapdelete otevře spojení na server LDAP, naváže je a zruší jednu nebo více položek. Je-li zadán alespoň jeden parametr dn, jsou zrušeny položky, které mají tato jednoznačná jména (Distinguis-hed Names, DN). Všechna dn jsou řetězce, které představují DN dle definice v RFC 1779. Nejsou--li zadány žádné parametry dn, jejich seznam se přečte ze standardního vstupu (nebo ze soubo-ru, je-li zadán parametr -f).

Zde je několik příkladů použití ldapdelete:
ldapdelete ‘cn=Luiz Malere,o=TUDelft,c=NL’ ldapdelete -v ‘cn=Rene van Leuken,o=TUDelft,c=NL’ -D ‘cn=Luiz Malere,o=TUDelft,c=NL’ -W

Volba -v znamená „upovídaný“ režim, volba -D znamená Binddn (dn, proti němuž se autentizu-
je) a volba -W znamená zadání hesla po vypsání promptu.ldapmodify – je shellové rozhraní ke knihovním voláním ldap_modify(3) a ldap_add(3). Tentonástroj je vhodný k modifikaci položek ve struktuře LDAP.
Stručný výtah volání vypadá takto (význam voleb viz manuálové stránky ldapmodify):

ldapmodify [-a] [-b] [-c] [-r] [-n] [-v] [-k] [-d debuglevel] [-D binddn] [-W] [-w passwd] [-h ldaphost] [-p ldapport] [-f file]

ldapadd [-b] [-c] [-r] [-n] [-v] [-k] [-K] [-d debuglevel] [-D binddn] [-w passwd] [-h ldaphost] [-p ldapport] [-f file]

ldapadd je implementována jako pevný odkaz na nástroj ldapmodify. Je-li zadán jako ldapadd, automaticky se zapne příznak -a (přidej novou položku). ldapmodify otevře spojení na server LDAP, naváže je a modifikuje nebo přidává položky. Informace o položkách se čtou ze standard-ního vstupu anebo v případě volby -f ze souboru.

Zde je několik příkladů použití ldapmodify:

Předpokládejme, že existuje soubor /tmp/entrymods a obsahuje: dn: cn=Modify Me, o=University of Michigan, c=US changetype: modify replace: mail mail: modme@terminator.rs.itd.umich.edu -add: title title: Grand Poobah -add: jpegPhoto jpegPhoto: /tmp/modme.jpeg -delete: description -

Příkaz:

ldapmodify -b -r -f /tmp/entrymods

nahradí obsah atributu „Modify Me“ položky mail hodnotou „*modme@terminator.rs.itd.umich.edu*“, přidá title typu „Grand Poobah“ a obsah souboru /tmp/modme.jpeg jako jpegPhoto a zcela odstra-ní atribut description.

Stejnou modifikaci jako shora lze provést pomocí staršího vstupního tvaru ldapmodify:

cn=Modify Me, o=University of Michigan, c=US mail=modme@terminator.rs.itd.umich.edu +title=Grand Poobah +jpegPhoto=/tmp/modme.jpeg -description
a příkazem:

ldapmodify -b -r -f /tmp/entrymods

Předpokládejme, že soubor /tmp/newentry existuje a obsahuje: dn: cn=Barbara Jensen, o=University of Michigan, c=US objectClass: person cn: Barbara Jensen cn: Babs Jensen sn: Jensen title: the world's most famous manager mail: bjensen@terminator.rs.itd.umich.edu uid: bjensen

Příkaz:

ldapadd -f /tmp/entrymods

přidá položku s dn: cn=Barbara Jensen, o=University of Michigan, c=US, pokud už není přítom-na. Když položka s tímto dn už existuje, příkaz ohlásí chybu a původní položku nepřepíše. Za předpokladu, že existuje soubor /tmp/newentry a obsahuje:

dn: cn=Barbara Jensen, o=University of Michigan, c=US changetype: delete

Příkaz:

ldapmodify -f /tmp/entrymods

odstraní položku Babs Jensen.Volba -f znamená soubor (čti modifikační informace ze souboru, nikoli ze standardního vstupu),-b znamená binární (hodnoty začínající lomítkem („/“) na vstupním souboru jsou považovány zabinární), -r znamená nahrazení (nahraď stávající hodnotu hodnotou implicitní).

Další informace a funkce

V této kapitole naleznete další informace a užitečné odkazy na témata jako autentizace, logy a kli-enti LDAP. Na konci kapitoly jsou doporučena velice pěkná URL a knihy o LDAP.

Migrační nástroje LDAP

Firma PADL Software Ltd. nabízí tzv. migrační nástroje LDAP jako skripty v jazyce Perl. Používajíse ke konverzi konfiguračních souborů do formátu LDIF. Doporučuji, abyste si licenční ujednánípřečetli ještě předtím, než je začnete používat, i když jsou volně šiřitelná. Máte-li v úmyslu pou-žívat server LDAP k autentizaci uživatelů, tyto nástroje vám mohou být velice k užitku. Lze jimizkonvergovat NIS nebo archív hesel do formátu LDIF, čímž se stanou kompatibilními se serveremLDAP. Jsou také vhodné pro migraci uživatelů, skupin, aliasů, počítačů, podsítí, sítí, protokolů,RPC a služeb z existujících služeb (NIS, nešifrované soubory a NetInfo) do formátu LDIF.

Migrační nástroje a informace o nich najdete na adrese *http://www.padl.com/tools.html.*

Balík obsahuje soubor README a jména skriptů jsou zvolena intuitivně. Nejprve je vhodnénahlédnout do souboru README a poté aplikovat skripty.Jinou adresou s migračními nástroji, kterou lze doporučit, je *http://dataconv.org/apps_ldap.html.*

Autentizace pomocí LDAP

Aby klient získal přístup ke službám LDAP, musí se nejdříve autentizovat. To znamená, že musí serveru LDAP říct, kdo získá přístup k datům, aby se server mohl rozhodovat, co který klient může vidět a co může dělat. Jestliže se klient autentizuje serveru úspěšně a server následně od tohoto klienta obdrží dotaz, ověří si, zda je klient oprávněn zadat takový dotaz. Tento proces se nazývá řízení přístupu.

V LDAP se autentizace provádí operací „navázání“ („bind“) spojení. Ldapv3 podporuje tři typy autentizace: anonymní,

jednoduchou a SASL. Klient, který pošle LDAP požadavek bez „navázání“ spojení, je považován za anonymního klienta. Jednoduchá autentizace spočívá v zaslání úplného DN klienta (uživatele) a hesla v nezašifrované podobě. Tento mechanismus je z hlediska bezpeč-nosti problematický, neboť heslo může na síti kdokoli přečíst. Přečtení lze zabránit tak, že jedno-duchý autentizační mechanismus proběhne v zašifrovaném kanále (např. SSL) za předpokladu, že jej podporuje server LDAP.

A konečně SASL (Simple Authentication and Security Layer, RFC 2222) specifikuje protokol typu výzva/odpověď, v němž jsou data mezi serverem a klientem vyměňována za účelem autentizace a ustavení bezpečné vrstvy, v níž probíhá následná komunikace. S použitím SASL může LDAP podporovat libovolný typ autentizace odsouhlasené mezi klientem a serverem. Balík Cyrus-SASL je k dispozici na adrese *http://asg.web.cmu.edu/sasl/sasl-library.html.*

Aby měl uživatel přístup k informacím z adresářového stromu, může server LDAP autentizovat uži-vatele i z jiných služeb (Sendmail, Login, Ftp atd.). To se provádí pomocí migračních informací specifických vzhledem k uživateli na serveru LDAP mechanismem, který se jmenuje PAM (Plug-gable Authentication Module). Autentizační modul pro LDAP je k dispozici v tarové podobě na adrese *http://www.padl.com/OSS/pam_ldap.html.*

Konfigurace SASL: Digest-MD5

Autentizaci LDAP-SASL provozuji prostřednictvím mechanismu DIGEST-MD5. K tomu je nutno provést přesně tyto kroky:

- Stažení SleepyCat 4.2.52, překlad a manuální vygenerování. Po stažení jsem se pouze řídil

  pokyny v souboru docs/index.html pod adresářem, v němž jsem rozbalil balík .tar.gz. Po rozbalení můžete dle pokynu spustit:

  root@rdnt03:/usr/local/BerkeleyDB.4.2/build_unix#../dist/configure root@rdnt03:/usr/local/BerkeleyDB.
  4.2/build_unix#make root@rdnt03:/usr/local/BerkeleyDB.4.2/build_unix#make install

- Stažení Cyrus SASL 2.1.17, rozbalení a další postup podle pokynů v dokumentu doc/install.html, pod adresářem, v němž jsem rozbalil balík .tar.gz. Musíte dát pozor na to, abyste spustili konfigurační skript se správnými vnějšími parametry:

  root@rdnt03:/usr/local/cyrus-sasl-2.1.17#env CPPFLAGS=”-I/usr/local/BerkeleyDB.4.2/include” LDFLAGS=”-L/usr/local/BerkeleyDB.
  4.2/lib” ./configure

Vnější parametry CPPFLAGS a LDFLAGS musí ukazovat na ty adresáře include a lib, ve kterých je nainstalována Berkeley BDB. Poté můžete dle pokynů spustit:

root@rdnt03:/usr/local/cyrus-sasl-2.1.17#make
root@rdnt03:/usr/local/cyrus-sasl-2.1.17#make install

root@rdnt03:/usr/local/cyrus-sasl-2.1.17#ln -s /usr/local/lib/sasl2
/usr/lib/sasl2

- Nakonec jsem (opět pomocí pokynů uvedených v tomto dokumentu) nainstaloval Open-LDAP 2.2.5. Pouze jsem spustil konfigurační skript jako při konfiguraci SASL:

  root@rdnt03:/usr/local/openldap-2.2.5#env CPPFLAGS=”
  I/usr/local/BerkeleyDB.4.2/include”
  LDFLAGS=”-L/usr/local/BerkeleyDB.4.2/lib” ./configure

- Pak jsem dle pokynů spustil:

  root@rdnt03:/usr/local/openldap-2.2.5#make dependroot@rdnt03:/usr/local/openldap-2.2.5#make
  root@rdnt03:/usr/local/openldap-2.2.5#make install

- Dále jsem vytvořil sasl pod databází:

  root@rdnt03:~# saslpasswd2 -c admin

Budete požádáni o heslo. Připomínám, že uživatelské jméno nemůže být DN (distinguis-hed name). Také musíte použít stejné heslo, které je uvedeno v položce admin v adresá-řovém stromě.
Ještě než spustíte démona slapd a otestujete autentizaci, musíte v souboru *slapd.conf* nastavit příkaz sasl-regexp. Můj soubor slapd.conf je uložen v adresáři /usr/local/etc/openldap:

sasl-regexp uid=(.*),cn=rdnt03,cn=DIGEST-MD5,cn=auth uid=$1,ou=People,o=Ever

Tento parametr má tvar: uid=<username>,cn=<realm>,cn=<mech>,cn=auth
Uživatelské jméno se vezme ze sasl a vloží se do vyhledávacího řetězce ldap místo $1; realm je považován za úplné jméno domény (fully qualified domain name, FQDN), avšak někdy, jako v mém případě, tomu tak není. Co je realm, najdeme pomocí:

```
root@rdnt03:~# sasldblistusers2
admin@rdnt03: userPassword
admin@rdnt03: cmusaslsecretOTP
```

- V mém případě je jako realm indikováno *rdnt03*. Pokud je to úplné jméno vaší domény, neměli byste mít žádný problém. Použil jsem následující soubor LDIF:

```
dn: o=Ever
o: Ever
description: Organization Root
objectClass: top
objectClass: organization


dn: ou=Staff, o=Ever ou: Staff description: These are privileged users that can interact with Organiza-tion products objectClass: top
objectClass: organizationalUnit

dn: ou=People, o=Ever
    ou: People
    objectClass: top
    objectClass: organizationalUnit


    dn: uid=admin, ou=Staff, o=Ever
    uid: admin
    cn: LDAP Adminstrator
    sn: admin
    userPassword: {SHA}5en6G6MezRroT3XKqkdPOmY/BfQ=
    objectClass: Top
    objectClass: Person
    objectClass: Organizationalperson
    objectClass: Inetorgperson


    dn: uid=admin,ou=People,o=Ever
    objectClass: top
    objectClass: person
    objectClass: organizationalPerson
    objectClass: inetOrgPerson
    userPassword: {SHA}5en6G6MezRroT3XKqkdPOmY/BfQ=
    displayName: admin
    mail: admin@eversystems.com.br
    uid: admin
    cn: Administrator
    sn: admin
```

  - Pomocí následujícího příkazu přidejte do svého adresáře LDAP položky:

```
slapadd -c -l Ever.ldif -f slapd.conf -v -d 256
```

  - Nyní spusťte démona *slapd* a pomocí ldapsearch zadejte dotaz:

```
root@rdnt03:~# ldapsearch -U admin@rdnt03 -b 'o=Ever' '(objectclass=*)' SASL/DIGEST-MD5 authentication started Please enter your
password: SASL username: admin@rdnt03 SASL SSF: 128 SASL installing layers ... Entries ...
```

A je to! Dáte-li přednost SASL se systémem Kerberos V nebo GSSAPI, zajímavý odkaz je na *http://www.openldap.org/doc/admin22/sasl.html.* Musíte však už mít zvládnutou instalaci a konfi-guraci knihovny SASL. Také je dobré se podívat na e-mailové konference na adrese *http://asg.web.cmu.edu/sasl/index.html#mailinglists.*

Grafické nástroje LDAP

Kldap je grafický klient LDAP napsaný pro KDE. Má velice zdařilé rozhraní, jehož pomocí může-te prohlížet všechny údaje o stromu ve svém adresáři. Na adrese *http://www.mountpoint.ch/ oliver/kldap/* si můžete prohlédnout některé obrazovky a stáhnout

si je.

KDirAdm je nástroj pro správu adresářů LDAP napsaný pro KDE Desktop Environment, verze 2 a pozdější. Jeho cílem je poskytnout veškerou funkcionalitu většiny komerčních nástrojů pro sprá-vu adresářů: *http://www.carillonis.com/kdiradm/. Directory Administrator* je nejrozšířenější aplikací pod GNOME pro správu uživatelů a skupin v Linuxu na adresářových serverech LDAP. Umožňuje vytvářet a rušit uživatele a skupiny a spra-vovat je pomocí údajů v adresářové knize, která je součástí aplikace. Dále řízení přístupu k jed-notlivým serverům a posílání pošty pomocí Sendmail: *http://diradmin.open-it.org/index.php.*

GQ je grafickým klientem LDAP s jednodušším rozhraním. Je napsán pro GNOME, avšak běží pod KDE, podobně jako i Kldap běží pod GNOME. Stáhnout si jej můžete z adresy *http://biot.com/gq/* a zde také o tomto klientovi naleznete další informace. *LDAP Browser/Editor* Tento nástroj je skutečně fantastický, nabízí kompletní správu a funkciona-litu pro prohlížení a vyhledávání. Vyzkoušejte ho, je na stránkách Ldap Browser na adrese

http://www.iit.edu/~gawojar/ldap/.

Logy

Pro vytváření logovacích souborů využívá slapd démona syslog. Implicitní kategorií syslogu je LOCAL4, avšak jsou povoleny hodnoty od LOCAL0, LOCAL1 až po LOCAL7 (kategorie LOCALx označují „lokální“ kategorie zpráv).

Generování logů je nutno aktivovat změnou v souboru syslog.conf, který je obvykle umístěnv adresáři /etc. Vytvořte například řádek:

```
local4.* /usr/adm/ldaplog
```

Příkaz pro nastavení syslogu využije implicitní kategorii LOCAL4. Pokud vám není jasná syntaxe tohoto řádku, nahlédněte do manuálových stránek syslog, syslog.conf a syslogd. Pomocí následujících voleb při spuštění slapd lze specifikovat úroveň vygenerovaných logů nebo změnit kategorii:

```
-s syslog-level
```

Tato volba říká slapd, na jaké úrovni mají být zapisovány ladicí příkazy do syslog. Úroveň udává závažnost zprávy a je dána jedním z klíčových slov z následujícího uspořádaného seznamu (od vyšších k nižším): emerg, alert, crit, err, warning, notice, info a debug.

Příklad: slapd -f myslapd.conf -s debug.

```
-l syslog-local-user
```

Vyberte kategorii syslogu. Hodnotami mohou být LOCAL0, LOCAL1 atd. až po LOCAL7. Implicit-ní je LOCAL4. Tato volba je nicméně přípustná pouze v systémech, které podporují lokální kate-gorie v syslogu.

Nyní se můžete podívat na vygenerované soubory s logy (v uvedeném příkladu je to /usr/adm/ldaplog). Mohou vám značně ulehčit řešení problémů s dotazy, aktualizacemi, nava-zováním spojení atd.

Odkazy

Zde jsou adresy URL, které obsahují velmi užitečné informace o LDAP. Z nich byl vytvořen tento návod, takže pokud byste po jeho přečtení potřebovali přesnější informace, pravděpodobně je naleznete zde:

University of Michigan LDAP (vývoj): *http://www.umich.edu/~dirsvcs/ldap/*
University of Michigan LDAP (dokumentace): *http://www.umich.edu/~dirsvcs/ldap/doc/*
OpenLDAP Administrator’s Guide (spřízněný dokument): *http://www.openldap.org/ doc/admin*
Linux Directory Service: *http://www.rage.net/ldap/*
Red Hat and LDAP: *http://www.redhat.com/docs/manuals/linux/RHL-9-Manual/ref-guide/ch-ldap.html*
Mandrake Linux – Using OpenLDAP for Authentication: *http://www.mandrakesecure.net/ en/docs/ldap-auth.php*
Integrating OpenLDAP with other Open Source projects: ftp://kalamazoolinux.org/pub/ pdf/ldapv3.pdf

Knihy

Nejoblíbenější a nejpotřebnější knihy o LDAP:

Mark Wilcox: Implementing LDAP
Howes, Smith: LDAP: Programming Directory-Enabled Applications with Lightweight Directory Access Protocol
Howes, Smith, Good: Understanding and Deploying LDAP Directory Servers

RFC

RFC (*http://www.rfc-editor.org/rfc/*) související s vývojem LDAP:

RFC 1558: A String Representation of LDAP Search Filters

RFC 1777: Lightweight Directory Access Protocol
RFC 1778: The String Representation of Standard Attribute Syntaxes
RFC 1779: A String Representation of Distinguished Names
RFC 1781: Using the OSI Directory to Achieve User Friendly Naming
RFC 1798: Connectionless LDAP
RFC 1823: The LDAP Application Programming Interface
RFC 1959: An LDAP URL Format
RFC 1960: A String Representation of LDAP Search Filters
RFC 2251: Lightweight Directory Access Protocol (v3)
RFC 2307: LDAP as a Network Information Service

Textové terminály

Úvod

V tomto dokumentu si vysvětlíme, co jsou terminály, jak pracují, jak je nainstalujete a zkonfigu-rujete, a naleznete zde i zmínku o opravách terminálů. Dokument může přijít vhod, zejména nemáte-li k terminálu příslušný manuál. I když jsou popisovány reálné terminály v systému Linux, řada informací je použitelná i pro emulované terminály a v jiných systémech než Linux.

Chcete-li učinit rychlý pokus o instalaci, nahlédněte do Quick Install, *http://tldp.org/HOWTO/ Text-Terminal-HOWTO-3.html#quick_install.*

Prohlášení

I když jsem se nikoho nepokusil úmyslně uvést v omyl, dokument může obsahovat řadu chyb. Naleznete-li nějaké, upozorněte mě na ně, prosím. Vzhledem k tomu, že jde o volně šiřitelný dokument, má se za to, že nenesu právní odpovědnost za žádnou chybu.

Nové verze tohoto návodu

Nové verze Návodu k textovým terminálům by měly vycházet zhruba každý rok. Budou na zrcad-lech LDP, kde je bude možno prohlížet i stahovat. Seznam zrcadel naleznete na *http://tldp.org/ mirrors.html.* Dokument bude dostupný v různých formátech. Datum vydání poslední verze lze rychle zjistit na *http://tldp.org/HOWTO/Text-Terminal-HOWTO.html.* Verze, kterou právě čtete, má číslo 1.38 a je z února 2006.

Příbuzné návody

Návody naleznete na nejbližším zrcadle (viz shora).

V Serial-HOWTO jsou informace o multiportových sériových kartách pro modemy i termi-nály. Jsou zde také obecné technické informace o sériových portech včetně odstraňování závad.
Low-Level Terminal Interface (*http://tldp.org/HOWTO/www.gnu.org/manual/glibc/ html_chapter/libc_12.html*) jako součást „GNU C Library Reference Manual“ (v dokumen-tačním balíku libc nebo gibc). Pokrývá detailně význam příkazů stty atd.
NCURSES-Programming-HOWTO
MacTerminal mini-HOWTO
Modem-HOWTO
Serial-Programming-HOWTO
NC mini-HOWTO
NCD-X-Terminal mini-HOWTO
XDM-and-X-Terminal mini-HOWTO
Connecting-X-Terminals-to-Linux-Mini-HOWTO
NCD-HOWTO
Thinclient-HOWTO
Xterminals-HOWTO
Xterm-Title-HOWTO (pouze pro změnu titulu okna)

Terminologie používaná v tomto dokumentu

Konfigurace znamená totéž co nastavení. Zatímco příkazy v Linuxu obsahují volby (označované symboly - nebo --), volba v širším slova smyslu znamená výběr různých typů. Instalace v širším slova smyslu zahrnuje nastavení (konfiguraci) softwaru i hardwaru. Tvrzení, o němž předpoklá-dám, že je pravdivé (ale nemusí být), končí dvěma otazníky (??). Máte-li v této věci jasno, dejte mi vědět.

Co je to terminál?

Reálný terminál se skládá z obrazovky a z klávesnice, které se používají ke vzdálené komunikaci s počítačem. Používají se, jako kdyby to byl osobní počítač, avšak terminál je od počítače vzdá-len (na druhém konci místnosti nebo na druhém konci světa). Programy se provádějí na počítači, avšak výsledky jsou zobrazovány na obrazovce terminálu. Výpočetní schopnost terminálu je rela-tivně nízká (jinak by to nebyl terminál, nýbrž počítač). Obecně je terminál omezen na schopnost zobrazovat to, co se mu

pošle (třeba včetně grafiky pokrývající celou obrazovku), a schopnost odesílat počítači to, co je napsáno na klávesnici.

Textový terminál pouze zobrazuje text bez obrázků. V dobách sálových počítačů od poloviny 70. do poloviny 80. let používala většina lidí ke komunikaci s počítači skutečné terminály. Pořizovali na nich programy, spouštěli je, psali na nich dokumenty, vydávali z nich příkazy k tisku atd. Ter-minál byl k počítači připojen (často nepřímo) kabelem. Název terminál vznikl tak, že terminály byly připojeny ke konci těchto kabelů. Některé textové terminály se nazývaly „grafické“, avšak vinou vysoké ceny paměti a omezené rychlosti sériových portů bylo rozlišení oproti dnešním stan-dardům nevalné a rychlost byla nízká.

Dnes už nejsou reálné terminály tak časté jako kdysi a většina lidí, kteří používají terminály, pou-žívají osobní počítač, který terminál pouze emuluje. Prakticky každý, kdo používá Linux, používá i terminálovou emulaci. Bez systému X Window se používá textové rozhraní (virtuální terminál). Říká se mu také rozhraní příkazového řádku. V systému X Window může mít jeden uživatel něko-lik terminálových oken (xterm, rxvt nebo zterm). Všechny emulují reálný terminál softwarově.

Reálný textový terminál je něco jiného než monitor, je vybaven jinou elektronikou. Často je připojen k sériovému portu počítače dlouhým kabelem. Narozdíl od monitoru, který bývá umístěn přímo u počítače, tedy může být terminál od počítače poměrně dost vzdálený. Zatímco monitor dostává obrázek z videokarty umístěné v počítači, obdoba takové karty je zabudovaná přímo v terminálu. Terminály ovšem často bývají monochromatické a mají jen jednoduchou grafiku, takže možnosti takové „videokarty“ jsou omezené. Většina terminálů také nemá myš. V síťové terminologii klient/server by se člověk mohl domnívat, že terminál je klientem a počítač je server. Terminálům se někdy říká „tenký klient“. Ve skutečnosti však nejde vůbec o „klienta“ a počítač není „server“. Jedinou „službou“, kterou počítač terminálu poskytuje, je zpracování jed-notlivých písmen napsaných na klávesnici a reakce, jaká by se dala očekávat od počítače. Termi-nál je v počítači v podstatě oknem, podobně jako jsou jimi monitor a klávesnice. Je možné, že jste v Linuxu už virtuální terminály někdy používali (stisknete levý Alt-F2 atd.). Reálný terminál je podobný terminálu virtuálnímu, avšak provozujete jej na vlastní obrazovce, místo abyste sdíleli obrazovku monitoru. Narozdíl od virtuálního terminálu na konzole (monitoru) se může k tomuto terminálu posadit někdo jiný a používat počítač současně s ostatními.

Typy terminálů

Hloupé terminály

Existuje celá řada protichůdných definic „hloupého terminálu“, avšak s postupem času hloupých terminálů přibývá. V tomto dokumentu jsou popsány především textové terminály, které na obra-zovce zobrazují pouze text. Dokument bychom mohli nazvat i „Návod k hloupým terminálům“. Nicméně, v některých časopiseckých článcích jsou všechny terminály nazývány hloupými, včetně těch, které nabízejí uživateli plné grafické rozhraní (GUI). Jsou-li hloupé všechny terminály, je zby-tečné je takto označovat, snad jedině s výjimkou, když chce prodejce zdůraznit rozdíl mezi ter-minálem a počítačem či něčím podobným. Pak ovšem dvojsmyslný výraz „hloupý terminál“ nabý-vá onoho druhého významu a nejde o klasifikaci typu terminálu.

Textové terminály

Po kabelu, který spojuje počítač s textovým terminálem, tečou data dvěma směry. Tok probíhá v bajtech (např. v ASCII), přičemž každý bajt obvykle představuje tisknutelný znak. Bajty napsa-né na klávesnici jdou do počítače a většina znaků jdoucích z počítače je zobrazena na obrazovce terminálu. Speciální řídicí bajty (nebo jejich posloupnosti) z počítače sdělují terminálu, kam má posunout kurzor, co má vymazat, kde začít a skončit s podtržením, s blikáním, s tučným písmem atd. Často existují stovky takových příkazů a některé terminály dokonce umějí měnit typ písma.

Při komunikaci jsou znaky (písmena) kódovány pomocí kódové tabulky s danou znakovou mno-žinou. Prvních 128 bajtů z celkových 256 je obvykle zakódováno v kódu ASCII. Terminály unixo-vých systémů jsou obvykle připojeny k počítači kabelem, který propojuje asynchronní sériové porty (RS-232-C = EIA-232-D) počítače a terminálu. Někdy může spojení tvořit modem, termináloový server apod.

„Textový terminál“ také někdy nazýváme „sériový monitor“, „sériová konzola“ (je-li používán jako konzola), „sériový terminál“, „hloupý terminál“, „znakově-buňkový terminál“, „znakový terminál“, „ASCII/ANSI terminál“, „asynchronní terminál“, „datový terminál“, „videoterminál“, „videodisplejo-vý terminál“ (VDT) nebo „zelený terminál“ (protože většina má zelený displej). Tyto názvy (zejmé-na pak „hloupý terminál“) se někdy používají ve významu emulovaný terminál na PC s řádkovým rozhraním, například v Linuxu. Dříve se také říkalo „videodisplejová jednotka“ (VDU), avšak tento název vylučuje klávesnici.

Terminály starých sálových počítačů IBM používaly výhradně „blokový režim“, mají jej ovšem i některé moderní terminály (i když ho příliš často nepoužívají). V blokovém režimu se zadané znaky pozdrží v paměti terminálu (a lze je dokonce měnit editorem vestavěným v terminále). Pak, po stisknutí odesílací klávesy (nebo nějaké podobné), se do počítače naráz odešle celý blok znaků (většinou právě jeden řádek). Linux nepodporuje blokový režim asi od konce roku 1998, viz kapitola „Blokový režim“.

Grafické možnosti textových terminálů

Mnoho textových terminálů umí zobrazit bitmapové obrázky, nikoli však barevně. Bohužel tvar s oblibou používaný na Internetu není podporován. Protokoly pro terminálovou grafiku jsou: Tek-tronix Vector Graphics, ReGIS (DEC), Sixel (DEC) a NAPLPS

(North American Presentation Level Protocol Syntax).

Obyčejné terminály umějí něco jako zobrazovat obrázky i bez bitových map. Šipku můžete vytvo-řit takto: <---, čtverečky takto: |__|, atd. Se speciálními grafickými znakovými množinami obsa-hujícími množství speciálních znaků jsou možnosti daleko větší. „Ascii grafiku“ však můžeme vytvářet i bez množiny grafických znaků. Pojem „grafický terminál“ obvykle znamená terminál, který umí zobrazit bitmapové obrázky. Tento termín se však někdy používá i v souvislosti s výhradně textovými terminály, neboť text je omezená forma grafiky.

Grafické (GUI) displeje

Existují dva základní typy grafických displejů: rastrový a vektorový (méně používaný). Rastrová grafika (bitmapová) spočívá v tom, že se na obrazovku do vodorovných řádků skládají body, které kreslí paprsek elektronů (anebo na ploché obrazovce jsou aktivovány pixely, resp. body). Disp-leje s vektorovou grafikou jsou určeny pro monochromatické obrazovky, které nejsou složeny z bodů. Využívají k tomu inteligentní elektroniku, kterou kreslí čáry a křivky pomocí paprsku elek-tronů, který lze přesouvat libovolným směrem (jako pero nebo tužku). Čistá vektorová grafika kreslí čáry ve vysoké kvalitě bez viditelných nespojitostí, avšak je málo používaná a drahá. Další podrobnosti viz *http://www.cca.org/vector/*. Rastrová grafika je prakticky univerzální jak na PC, tak i na terminálech. Obrázky zakódované ve vektorové grafice nemohou být na PC kresleny jako souvislé čáry, neboť to nezvládne elektronika, avšak mohou být převedeny do rastrové grafiky (s určitou ztrátou kvality obrázku).

Tencí klienti (terminály?)

Vzhledem k tomu, že „tencí klienti“ nejsou textové terminály, tento návod jenom poskytuje jejich krátký přehled. Existují jiné, detailnější návody, viz kapitola „Podobné návody“. Tencí klienti jsou počítače v minimální sestavě, které se chovají jako terminály. Protože v textových terminálech (kromě těch nejstarších) jsou vestavěné operační systémy, jsou to vlastně také počítače. Tencí kli-enti musí mít větší výpočetní kapacitu. Na rozdíl od textových terminálů mají tencí klienti moder-ní vysokorychlostní grafické uživatelské rozhraní (Graphic User Interface, GUI). Pro svoji činnost potřebují počítače (servery) s vysokým výkonem. V případě terminálu, který je skutečně klientem, bude veškeré zpracování a ukládání dat na disk provádět server. Opačným extrémem je, když vět-šinu z této činnosti provádí tenký klient, avšak některé administrativní činnosti přece jen zůstanou na serveru. Takový klient ovšem není „tenký“, takže správně by měl být nazýván „tlustým klien-tem“.

Takoví klienti mohou být vytvořeni z běžného PC pomocí softwaru nebo mohou být samostatnou hardwarovou jednotkou. Ta ovšem nejspíš bude mít normální monitor jako PC a nějakou bedýn-ku s kouskem počítačového hardwaru. V Linuxu se zřejmě jako klient používá PC.

Existuje i názor, že textové terminály jsou rovněž tenkými klienty, což ovšem nejsou, neboť neod-povídají modelu klient/server. Je pravda, že připojení terminálu prostřednictvím telnetu poněkud připomíná model klient/server tím, že telnet je prostředkem pro přenos dat. Avšak vztah mezi tex-tovým terminálem a počítačem není vztahem mezi klientem a serverem. Textový terminál je pouze jiným prostředkem, který zajišťuje přístup k počítači, podobně jako monitor s klávesnicí. Totéž by bylo možné tvrdit o tenkém klientovi s tím, že vztah klient/server zajišťuje pouze přenos dat.

Tenký klient je tedy vlastně terminálem. Má grafické rozhraní s myší, což vypadá, jako kdybyste pracovali přímo s počítačem. Skutečně pracujete, ale počítač může být hodně daleko a může jej současně používat mnoho dalších uživatelů. Komunikace probíhá vysokorychlostním síťovým kabelem, nebo dokonce po Internetu. Někteří tencí klienti navíc umějí emulovat textový terminál a mají pro tento účel konektor na sériový port anebo rozhraní USB.

Existuje mnoho různých typů tenkých klientů. Jedním z nich je „window-terminál“, který běží pod MS servery (a softwarem). Jiným typem je „síťový počítač“, z hlediska platformy neutrální. Zna-mená to, že by měl pracovat jak s MS Windows, tak i s Linuxem, avšak starší modely mohou mít v Linuxu problémy, neboť tento systém komunikuje v protokolu X Window. Viz kapitolu „Tencí klienti a NC pod Linuxem“.

Terminály MS Window

To jsou skutečně terminály, neboť veškeré výpočty probíhají na serveru se systémem Windows. Říká se jim také „terminály na bázi Window“ (Window-based Terminals, WBT). Podobají se počí-tači, neboť je na nich často provozován vestavěný operační systém, např. Linux nebo CE, NT, resp. XP firmy Microsoft. Bývají uloženy ve vnější (flash) paměti a je možno je aktualizovat. Běžné PC tedy lze používat jako klienta (v některých případech i linuxové PC) s příslušným softwarem. Ně-kteří klienti podporují X Window (z linuxového serveru) a někteří emulují textové terminály. Na řadě tzv. „síťových počítačů“ může být také provozován systém X Window, o čemž se zmíníme v následující kapitole.

Server pro tyto klienty bývá obvykle vybaven MS Terminal Services (pro servery Windows 2000). Jeho předchůdcem byl Windows NT Terminal Server Edition (od poloviny roku 1998 pod kódo-vým označením „Hydra“). MS používá RDP (Remote Desktop Protocol) založený na protokolu ITU T.120. Existuje navíc i volitelný protokol ICA (rozšířený), který umí spolupracovat s RDP.

Ještě před tím existoval modifikovaný Windows NT 3.51 (1995), nazývaný „WinFrame“, od Citri-xu, který používat vlastní (proprietární) protokol ICA (Independent Computing Architecture). Cit-rix zůstal na scéně i poté, co MS přišel s vlastním

terminálovým serverem. Vytvořil software Meta-Frame (dříve pICAsso) jako dodatek k systému MS Terminal Server (nebo Services) tak, aby pod-poroval terminály na bázi ICA a zajišťoval i další funkce. Než se do toho vložil Microsoft, existo-valy i jiné proprietární terminálové systémy schopné zobrazit grafické rozhraní MS Windows, avšak později všechny přešly na podporu systémů firmy Microsoft.

PC s Linuxem lze prostřednictvím „volného“ (pouze však co do ceny) proprietárního klientského softwaru ICA od Citrixu (Citrix Systems, Inc. – *http://www.citrix.com/download/unix-downloads.asp*) upravit na klientský terminál na bázi ICA. MS bohužel pro obsluhu klientů vyžaduje zakoupení licence, a to i tehdy, když všichni klienti běží pod Linuxem. Chcete-li tedy na softwaru ušetřit tím, že používáte Linux, musí to být čistě linuxové řešení, a to jak v případě klienta, tak i serveru s pro-tokolem X-Window, který je zdarma.

Říká se tomu někdy „síťové výpočty“, neboť terminály a servery jsou navzájem propojeny po Inter-netu (např. běžná síť TCP/IP používaná jak v Linuxu, tak i v MS Windows). Síťové počítače jsou něco jiného, viz dále.

Síové počítače (NC)

Není to ani čistě počítač, ani čistě terminál, je to něco mezi tím. Jedním z typů síťového počítače (NC) je počítač bez pevného disku. Operační systém je mu poslán po síti. NC mají plnou grafiku a využívají služeb serveru. Od terminálů se poněkud liší, neboť některé programy (nebo dokon-ce většina), jež provozují, jsou zpracovávány vlastním procesorem. Primární funkcí terminálů je provozovat internetový vyhledávač, proto jim nelze posílat ke zpracování applety v Javě. Mnoho NC podporuje systém X Window, takže pro jejich podporu lze použít linuxový server. Můžeme jej nazývat „linuxovým terminálovým serverem“. IBM nazývala svoje NC „NetStation“, nyní je však nazývá „NetVista“. Měly by fungovat v sítích typu intranet a samy pracují pod operačním systé-mem Linux.

Firma Wintel přišla s „NetPC“, který je narozdíl od předchozích NC téměř úplným osobním počí-tačem. Nemá však vyměnitelné disky, aby si uživatelé nemohli instalovat vlastní software nebo pořizovat kopie čehokoli.

Tencí klienti a NC pod Linuxem

Existuje tzv. „Linux Terminal Server Project“ (LTSP nebo ltsp), který umožňuje používat Linux jako server pro bezdiskové tenké klienty. Je vybaven systémem X Window a aplikace běží na něm. Lze jej však nastavit i tak, že některé nebo všechny aplikace běží na „terminále“. Viz
http://www.ltsp.org/.

„Terminál“ v LTSP je skutečně tenkým (nebo tlustým) klientem. Lze na něm provozovat sezení tel-net, může se tedy chovat jako textový terminál. V hlavních linuxových distribucích naleznete balí-ček lts pro LTSP.

Máte-li jen několik „terminálů“, mohou pracovat bez ltsp. Jakmile jich ale máte mnoho, software ltsp je nezbytný. Chcete-li tedy provozovat jako bezdiskového klienta nějaké starší PC nebo něco podobného, musíte použít ltsp. Pracuje to perfektně i na systémech s více než stovkou pracovních stanic jako tenkých klientů.

Jsou-li běžné počítače propojeny do sítě, můžete spouštět programy na jiném počítači pomocí NFS (Network File System). Takový program posílá zprávy po síti, takže se jeví, jako kdyby běžel na lokálním počítači. Ve skutečnosti však běží na jiném počítači připojeném k síti. Funguje to i s X Window, takže na grafickém rozhraní můžete vidět obrázky vytvořené na jiném počítači.

V Linuxu může být počítač v bezdiskové variantě a spouštět se po síti. Speciální software pro tento účel naleznete opět na stránkách „Terminal Server Project“ a stručný přehled na Network-boot-HOWTO. Starší dokumenty na toto téma jsou Diskless-HOWTO a Diskless-root-NFS-HOWTO. Pokud tedy používáte bezdiskový počítač s NFS, můžete zpracovávat programy na jiných počíta-čích (na serveru). Trochu je to podobné síťovému počítači (NC). Není to však přímo NC, je pouze emulován určitý typ NC. Říká se mu někdy pouze „terminál“, což v určitém smyslu skutečně je.

Máte-li tedy staré PC s ethernetovou kartou (NIC), můžete je používat jako síťový počítač. Jedním z možných zdrojů informací je Thinclient-HOWTO. Dokonce ani když PC nemá NIC (tedy síťovou kartu), lze je použít k emulaci textového terminálu, viz kapitola „Emulace terminálů“.

Existuje i spousta skutečných NC, které budou fungovat s linuxovým serverem. Některé i samy provozují Linux. Než vzrostla obliba Linuxu, síťové počítače jej neprovozovaly, a musely tedy mít jiné operační systémy. I takové NC mohou spolupracovat s linuxovým serverem. V takovém pří-padě je nelinuxový operační systém uložen na serveru jako běžný soubor a NC po spuštění pošle linuxovému serveru zprávu, v níž si tento soubor vyžádá. Server jej po síti pošle na NC a ten se spustí.
Na linuxovém serveru běží jak NFS, tak i X Window, oba jsou podporovány NC. To umožňujepracovat s NC jako s X Window terminálem.Pro provozování Linuxu na některých typech NC existují návody:

JavaStation-HOWTO (Sun),
NC-HOWTO (IBM NetStation),
NCD mini-HOWTO (NCD-ThinSTAR),
NCD-X-Terminal mini-HOWTO,
XDM-and-X-Terminal mini-HOWTO.

Hardwarové schéma

U tenkých klientů existují 3 různé druhy hardwarového uspořádání. První typ pouze používá PC jako tenkého klienta a emuluje tlustého klienta. Ve skutečnosti to není tenký klient, ale chová se tak. Druhý typ vypadá jako textový terminál, tedy jako monitor s jedním konektorem na kláves-nici a s jedním na síťový kabel. Je to jednoúčelový tenký klient, který nelze použít k ničemu jiné-mu. Třetí typ vypadá jako malinký počítač. Je vybaven standardním PC monitorem a klávesnicí, které se zasouvají do malého boxu, což je vlastně „tenký" počítač a tvoří rozhraní mezi monito-rem a klávesnicí na jedné straně a sítí na druhé straně.

Historie a budoucnost

Příznivci síťových počítačů a příbuzných window-terminálů měli v plánu, že tyto terminály brzy nahradí miliony osobních počítačů. V roce 1998 bylo na celém světě prodáno kolem 700 tis. ten-kých klientů (z nichž asi 27 % bylo NC). V roce 1999 prodej poklesl na 600 tis., avšak v roce 2000 opět vzrostl na 900 tis. (přičemž předpověď byla 1,3 mil. kusů). V roce 2001 se prodalo 1,09 mil. a předpověď na rok 2002 zněla na 1,4 mil. kusů.

Na trhu stále vedou (rok 2003) servery firmy Microsoft. Klienti mohou pracovat pod systémem Linux, i když za každého takového klienta je nutno zaplatit Microsoftu licenční poplatek. Půdu pod nohama tak získávají volně šiřitelné, čistě linuxové systémy.

Hlavním důvodem, proč růst prodeje zaostával za předpověďmi, byl pokles cen osobních počíta-čů v posledních letech, které tak nejsou o mnoho dražší než tencí klienti. Je nicméně otázkou, zda není lepší tenký klient ve srovnatelné ceně s osobním počítačem, vezmeme-li v úvahu nižší nákla-dy na jejich správu a údržbu. Tencí klienti někdy nahrazují textové terminály místo PC.

Emulace na PC

Vzhledem k tomu, že osobní počítač má obrazovku a klávesnici (podobně jako terminál), avšak disponuje mnohem větší výpočetní kapacitou, je možné využít její část na zajištění funkcí texto-vého terminálu. Tomu se říká „emulace terminálu" a emulují se obvykle textové terminály. Viz kapitola „Emulace terminálů".

Rychlá instalace

Jde o rychlé nainstalování terminálu bez nutnosti procházet procedurami nastavování terminálu i počítače popsanými v kapitole „Nastavení". Je-li terminál nastaven vzhledem k počítači nekom-patibilně, terminál po rychlé instalaci pravděpodobně nebude pracovat korektně. Pokud byste některé části v této kapitole nerozuměli, potřebné informace určitě naleznete v jiných částech tohoto dokumentu.

Máte-li v úmyslu instalovat terminál, nahlédněte nejdříve do /etc/termcap nebo terminfo.src, kde naleznete příslušnou položku. Zjistěte, ke kterému sériovému portu bude terminál připojen a jaké má označení tty (např. ttyS1 nebo tts/1, viz kapitola „Jména zařízení"). Jako uživatel root změňte soubor /etc/inittab tak, že za příkazy getty přidáte další příkaz getty, jehož tvar závisí na tom, jaký program getty používáte. Nejjednodušší je agetty (v distribuci Debian se nazý-vá pouze getty), nemá žádný konfigurační soubor. Nahlédněte i do manuálových stránek getty. Parametry getty získáte z databáze terminfo (nebo termcap) – hledejte podle jména terminálu, například vt100. Zadejte přenosovou rychlost (baud-rate), kterou terminál podporuje. Nastavíte-li ji příliš vysokou, pomoc naleznete v kapitole „Řízení přenosu".

Pak fyzicky připojte hlavní sériový port terminálu ke zvolenému sériovému portu počítače pří-slušným kabelem (null modemem) a zapněte terminál. Příliš nespoléhejte na to, že kabely z výro-by budou zapojeny správně. Přesvědčte se, zda jste nastavili stejnou přenosovou rychlost, jako jste uvedli v příkazy getty, a že „datových bitů" je 8. Na konzole počítače pak zadejte init q, aby se uplatnily změny provedené v souboru inittab. Na terminále byste měli nyní vidět přihlašovací prompt (výzva). Nebude-li tam, stiskněte klávesu return. Nebude-li terminál ani pak fungovat, pokračujte ve čtení tohoto dokumentu a podívejte se i do části odstraňování závad.

Proč používat terminál?

Úvod k této kapitole

Počítače jsou dnes už tak výkonné, že jedno PC většinou stačí obsloužit několik uživatelů sou-časně, zejména když počítač příliš nezatěžují, tj. například editují, pořizují data apod. K jednomu osobnímu (nebo jinému) počítači můžeme modemem nebo napřímo připojit několik terminálů. Je třeba mít počítač vybaven víceuživatelským systémem, což je například Linux, aby každý uživatel mohl využívat počítač nezávisle. Tento způsob se dříve nazýval „sdílení času", což ale dnes už není zcela vhodný výraz, neboť „distribuované" výpočty po síti jsou také svým způsobem sdíle-ním času. Lépe je označovat tento způsob práce jako „centralizované" výpočty. Centrální počítač však může být po síti připojen ke zbytku světa, uživatelé tedy mohou posílat e-maily, prohlížet Internet pomocí linuxových prohlížečů atd. Takže ani tento počítač nelze nazvat „centrálním".

Terminály se dříve zřídkakdy používaly s osobními počítači, neboť oblíbené operační systémy, které se na nich provozovaly

(Windows, DOS a Mac), nebyly až do roku 1998 víceuživatelské (první byl MS Windows NT) a podpora terminálů byla špatná. Nyní, když Linux je volně přístupný víceuživatelský systém pro osobní počítače, používání terminálů je snáze uskutečnitelné. Nevýhodou je, že textové terminály nejsou dostatečně inteligentní na to, aby podporovaly grafické rozhraní (GUI), jejichž použití se na současných počítačích běžně předpokládá.

Nižší cena hardwaru?

Když ještě byly počítače (i osobní) drahé, nižší ceny hardwaru byly výraznou výhodou terminálů. Dnes, když už jsou PC levné, se úspora jeví jako problematická. Přečtěte si, co jsem napsal před několika lety, když ještě byly PC drahé. Platí to dodnes, avšak tvrzení už nemá takový význam.

Používá-li počítač několik lidí současně, k zajištění stejné úrovně služeb je potřeba méně hard-waru. Jedna úspora spočívá ve sdílení kódu. Podobně jako jsou sdíleny knihovny, sdílejí se i apli-kační soubory na pevném disku (i když uživatelé provozují různé programy, používají řadu spo-lečných funkcí). Jiným typem úspory je lepší rozložení zatížení počítače. Pracuje-li na počítači pouze jeden uživatel, hardware běží po většinu času naprázdno, neboť člověk píše pomalu, pře-mýšlí, mluví a často odchází od stolu. Pracuje-li na počítači více lidí, je lépe využíván strojový čas, který by se jinak promarnil.

Úspory jsou značné. Lze (statisticky) odhadnout, že pro 9 osob (8 terminálů a jedna konzola) potřebuje sdílený počítač pouze trojnásobnou kapacitu (paměti, diskové paměti, základní jednot-ky atd.) oproti PC s jedním uživatelem. U sdíleného systému tedy stojí hardware pro výpočty na jednoho uživatele zhruba 1/3. Cena zobrazovacího hardwaru (displeje, klávesnice, videoelektro niky atd.) je však zhruba stejná. Terminály navíc musí mít počítač se sériovými porty.Aby bylo srovnání provedeno seriózně, terminály by měly mít stejné možnosti jako PC monitory.Bohužel, barevné grafické terminály pro Linux (X Window) s vysokorychlostní komunikací stojítéměř tolik jako PC a úspora je menší, pokud je vůbec nějaká. V případě textových terminálů jeovšem úspora prokazatelná, zejména když používáte levnější terminály.

Kontrola softwaru

U centralizovaných výpočtů proběhne instalace softwaru (a aktualizací) pouze na jediném počíta-či. Správu a konfiguraci nainstalovaného softwaru provádí většinou jedna osoba. Provádí-li tyto činnosti kvalifikovaně a zná-li dobře potřeby a preference uživatelů, přináší to s sebou značné výhody. Správce může uživatele omezovat v hraní her a surfování po Internetu tím, že nenainsta-luje příslušný software (anebo k němu omezí přístup). Výhodnost centralizovaného přístupu ke správě ovšem závisí na konkrétní situaci.

Hardwarové aktualizace

Aktualizace hardwaru se týká pouze jednoho počítače. Ušetří se tím práce s instalací. I když cena hardwarové aktualizace centrálního počítače bývá vyšší než cena aktualizace jednoho osobního počítače (to proto, že centrální počítač musí mít vyšší výkon), celková cena ve srovnání s aktualizací všech PC používaných jako terminály bude výrazně nižší.

Další výhody terminálů

Nepřítomnost hluku ventilátorů a diskových mechanik (pokud ovšem neemulujete terminál pomocí PC).
Uživatelé terminálů mohou sdílet data a soubory a mohou si navzájem posílat e-maily. Uspořádání je podobné jako v lokální síti.

Hlavní nevýhody terminálů

Textové terminály nemají vysokorychlostní grafické displeje (nebo grafiku s vysokým roz-lišením), i když pro kreslení čtverečků apod. mohou využívat grafické znakové množiny. Tím je omezen i výběr programů, které se mohou na terminálech používat.
Při výpadku centrálního počítače jsou terminály samostatně nepoužitelné (ledaže by byly ihned přepojeny na záložní počítač).

Jsou už textové terminály překonané?

Textové terminály jsou už technicky zastaralé, neboť za mírně vyšší cenu si můžete pořídit inteli-gentní terminál (s kvalitním displejem). To nebylo dříve možné, neboť například kolem roku 1980 činila cena paměti tisíce dolarů za megabajt. Nyní, když jsou paměti a procesory levné, lze poří-dit terminál s grafickým rozhraním za cenu o 10 až 20 % vyšší než běžný textový terminál. Vzhle-dem k tomu, že PC může terminál emulovat, emulátory terminálů jsou běžně k dispozici.

Příčiny toho, že textové terminály ještě nejsou zcela překonané, jsou:

Monochromatické terminály mají lepší rozlišení znaků než monitory v textovém režimu.
Uživatelé často nepotřebují obrazovku s plnou grafikou.
Vzhledem k tomu, že textové terminály (ve srovnání s GUI terminály) spotřebovávají méně prostředků centrálního počítače, může jich být k jednomu PC připojeno mnohem více.

Jak pracují terminály v Linuxu (přehled)

Viz také kapitola „Podrobnosti o práci terminálů“.

Jména zařízení

Všechny terminály jsou připojeny k sériovým portům centrálního počítače (často jako PC). Porty mají jména a čísla. Několik prvních: tts/0 (nebo ttyS0), tts/1 (nebo ttyS1), tts/2 (nebo ttyS2) atd.

Tato jména jsou současně jmény speciálních souborů v adresáři /dev (jako „device“, tedy zaříze-ní). Soubor tts/0 (resp. ttyS0) odpovídá COM1 v DOS a Windows, tts/1 (nebo ttyS1) je COM2 atd. Podrobnosti o těchto a podobných „zařízeních“ viz kapitola „Speciální terminálové sou-bory“.

Přihlášení/odhlášení

Ihned po zavedení systému spustí centrální počítač program getty, který spustí „přihlašovací“ pro-gram pro přihlašování účastníků. Viz kapitola „Getty (v /etc/inittab)“. Na obrazovce se objeví prompt „login:“ a účastníci se mohou po zadání hesla přihlásit. Od té chvíle mají přístup k cent-rálnímu počítači. Když chtějí ukončit činnost, odhlásí se a vypnou terminál. Popis omezení včet-ně přihlašování uživatele root viz kapitola „Omezení při přihlašování“.

Poloviční a plný duplex

Když pozorujete někoho, jak píše na terminálu, zadávaná písmena se současně objevují na obrazovce. Člověk by se mohl naivně domnívat, že zadávané znaky se posílají z klávesnice přímo na obrazovku a současně do počítače (jakoby poloduplexem, viz násl. odstavec). Ve skutečnosti jde znak z klávesnice přímo pouze do počítače, a teprve odtud je poslán zpět na terminál (plný duplex). Písmena se neposílají zpět pouze v některých případech (např. heslo nebo zkrácené příkazy editoru).

Plný duplex znamená, že existují dvě jednosměrná komunikační spojení. Plný duplex je pro terminál normou. Poloduplex je polovinou plného duplexu, tj. že existuje pouze jedno jednosměrné spojení, které sdílejí oba směry, a v daném okamžiku je možno použít pouze jeden směr. V takovém případě počítač není schopen opisovat zadávané znaky (a posílat je), takže terminál musí posílat každý znak i přímo na obrazovku. Některé terminály mají poloduplexní režim, avšak používá se velice zřídka.

Pamě terminálů

Pokud nebude obrázek na obrazovce obnovován paprskem elektronů vysílaným na stínítko, téměř okamžitě zmizí. Vzhledem k tomu, že text poslaný na terminál musí na obrazovce zůstat, je nutno jej uložit do paměti mikroprocesoru terminálu a paprsek elektronů jej musí na stínítko opakovaně kreslit (řekněme 60x za sekundu), aby se ze stínítka neztratil. Viz kapitola „Podrobnosti o paměti terminálu“.

Příkazy pro terminál

Terminál je řízen z počítače, který nejen posílá text ke zobrazení, nýbrž také posílá terminálu pří-kazy, které jsou prováděny. Jsou to řídicí kódy (bajty) a řídicí sekvence („escape sequences“). Například řídicí kód CR (carriage return, návrat vozíku) přesune kurzor na levý okraj obrazovky. Určitá řídicí sekvence (několik bajtů, z nichž první je řídicí kód „escape“) může přemístit kurzor na obrazovce na místo určené parametry umístěnými uvnitř řídicí sekvence.

Nejstarší terminály měly pouze několik takových příkazů, zatímco současné jich mají stovky. Vzhled displeje se může na různých místech měnit: jas, mat, podtržení, blikání nebo inverze. Když někdo zmáčkne klávesu, reproduktor může „cvaknout“ nebo pípnout, když dojde k chybě. Funkč-ní klávesy mohou být naprogramovány. Mohou existovat různé fonty. Displej může rolovat naho-ru a dolů. Na displeji lze určité oblasti vymazat. Mohou existovat různé druhy řízení toku dat, když například zasílaná data přicházejí na displej rychleji, než mohou být zpracována. Je jich mnohem a mnohem víc, jak se dozvíte z pokročilejších terminálových manuálů nebo z kapitoly „Seznam řídicích posloupností“.

Chybějící standardy řeší terminfo

Zatímco terminály vyráběné pro prodej v USA používají pro abecedu stejný kód ASCII (kromě ter-minálů IBM, které používají EBCDIC), bohužel nepoužívají stejné řídicí sekvence. To se stávalo dokonce i poté, co byly ustaveny různé standardy ANSI (a ISO). To proto, že nebyly vytvářeny s dostatečným předstihem. Navíc, starší terminály postrádaly možnosti novějších terminálů, což působilo problémy. Počítač například pošle terminálu řídicí sekvenci, aby byla obrazovka rozdě-lena na dvě okna určité velikosti, aniž by si uvědomil, že terminál toho není schopen.

K odstranění těchto problémů byla vytvořena databáze „termcap“ (ze slov „terminal capabilities“, schopnosti terminálů). Později byla termcap nahrazena databází „terminfo“. Tyto databáze jsou uloženy v určitých souborech na počítači a pro každý model terminálu mají samostatnou část nebo samostatný soubor. Pro každý model (např. VT100) je k dispozici seznam funkcí včetně řídicích sekvencí. Například blink=\E5m znamená, že terminál rozblikáte tak, že na něj pošlete Escape 5 m. Další podrobnosti viz

kapitola „Termcap a Terminfo“. Aplikační programy mohou tuto data-bázi používat prostřednictvím určitých funkcí knihovny C. Jedna z velkých skupin těchto progra-mů (přes 200) se jmenuje „ncurses“ a v manuálových stránkách stejnojmenného vývojového balí-ku je naleznete rovněž pod stejným označením – ncurses. Existuje i Návod k programování NCURSES (*http://www.tldp.org/HOWTO/NCURSES-Programming-HOWTO/index.html*).

Rozhraní

Vnější proměnná TERM obsahuje typ terminálu, o němž se Linux domnívá, že jej používáte. Vět-šina aplikačních programů ji používá pro hledání funkcí v databázi terminfo, takže musí být nasta-vena správně. K vytvoření správného rozhraní je toho ovšem potřeba trochu víc než jen znalost funkcí terminálu.

K tomu, aby mohl terminál přijímat tok bajtů z počítače, musí být nastaven na stejnou přenoso-vou rychlost, jakou jsou posílány z počítače. Je-li terminál nastaven na příjem 19 200 baudů a počí-tač posílá znaky rychlostí 9 600 baudů, na obrazovce pravděpodobně uvidíte jen zmatek nebo vůbec nic. Přenosovou rychlost terminálu (jakož i jiné funkce) vyberete v menu „set-up“ na ter-minále. Většina terminálů má v menu „set-up“ na vybranou mnoho voleb (viz kapitola „Nastavo-vání terminálů“). Volby mají i sériové porty počítačů a je nutno je nastavit tak, aby byla zachová-na kompatibilita.

Emulace

Většina dnešních terminálů má několik emulací (charakteristik, resp. „terminálových režimů“). Čísla modelů terminálů, která původně používala firma DEC (Digital Equipment Corporation, nyní Compaq), začínají od VT (tj. VT100). Mnoho dalších terminálů, které nejsou VT100, může emulo-vat VT100. Hlavním výrobcem terminálů je firma Wyse a většina jejich terminálů je schopna emu-lovat různé terminály DEC, např. VT100 a VT220. Chcete-li tedy používat řekněme terminál VT320, můžete si buďto pořídit skutečný terminál s charakteristikou VT320 nebo si vybrat některý jiný ter-minál, který umí emulovat VT320.

„Původní“ (nativní) charakteristika obvykle mívá více funkcí, takže i když ostatní kritéria jsou srov-natelná, je nejlépe používat terminál s „původní“ charakteristikou. Vzhledem k tomu, že linuxová konzola emuluje VT102, můžete si pořídit terminál, který jej emuluje (anebo něco podobného, např. VT100). Tím se přesvědčíte o tom, že některé programy, které neovládají terminály správně, budou na vašem terminále pracovat přece jen dobře. Když některé programy nenajdou terminfo s údaji o vašem terminálu (anebo nenajdou určité funkce), budou mít za to, že používáte VT102. Důvodem takového selhání tedy může být skutečnost, že nemáte k dispozici správný soubor term-info, což je chyba na vaší straně. Chyby způsobené a ohlášené „původním“ terminálem vám pomohou nalézt chyby, na něž byste třeba jinak vůbec nepřišli.

Nejběžnějším typem emulace je použití PC jako terminálu vt100 (nebo podobného). Emulaci pro-vádějí programy zavedené do paměti PC. V Linuxu (pokud nejste v X Window) monitor PC (nazý-vaný konzola) emuluje terminál typu „Linux“ (podobný vt100). Dokonce některá okna v X Win-dow emulují terminály, viz kapitola „Emulace terminálů“.

Konzola

Na osobním počítači je monitor konzolou. Emuluje terminál typu „Linux“. Lze se na něm přihlásit jako na virtuálním terminálu, jak si v tomto dokumentu ještě ukážeme. O spouštění a ukončová-ní systému dostává konzola zprávy od jádra systému. Zprávy, jež obvykle jdou na konzolu, lze přesměrovat na terminál. Dříve bylo nutno provést záplatu jádra, od verze jádra 2.2 lze přesmě-rování uskutečnit prostřednictvím „make config“. Viz kapitola „Jak se stane terminál konzolou“.

Speciální soubory terminálů (např. /dev/tty)

Označení „tty“ je zkratkou slova „teletype“ (dálnopis). Prvními terminály byly dálnopisy (neboli psací stroje ovládané na dálku). Viz podkapitola „Dálnopisy“. Seznam linuxových zařízení (obsah adresáře /dev) naleznete v „Linux Allocated Devices (Zařízení používaná v Linuxu)“ ve zdrojo-vých textech jádra. Seznam neobsahuje žádný návod k použití jednotlivých zařízení, pouze je několika slovy uvedeno, k čemu je zařízení vhodné.

Sériový port jako terminál

Počítač považuje sériové porty za „zařízení“. Někdy je nazýváme terminálová zařízení, neboť v jednu dobu byly terminály nejobvyklejším využitím sériových portů. Ke každému sériovému portu existuje v adresáři /dev speciální soubor. Ve světě DOS/Windows se speciální soubor /dev/tts/0 (nebo /dev/ttyS0) pro sériový port nazývá COM1. Zápis staršího typu ttyS0 vsou-borovém systému zařízení je nahrazen zápisem tts/0. Starší notaci budeme ovšem používat poměrně často, je dosud dost rozšířená a funguje (prostřednictvím symbolických odkazů) i v nových souborových systémech.
Text na terminál můžete poslat například tak, že přesměrujete standardní výstup příkazu na někte-rém příkazovém řádku do příslušného speciálního souboru. Napíšete-li za prompt na řádku napří-klad echo test > /dev/ttyS1 a máte-li oprávnění k zápisu do /dev/ttyS1, na terminál ttyS1 (COM2) se pošle slovo „test“. Podobně můžete například příkazem cat my_file > /dev/ttyS0 poslat obsah souboru my_file na COM1 (ttyS0).

Pseudoterminály

Pseudoterminály jsou dvojice zařízení, např. /dev/pty/m3 a /dev/pty/s3 (anebo /dev/ptyp3 a /dev/ttyp3, nepoužíváte-li souborový systém zařízení). S žádným z nich není spojeno žádné fyzické zařízení, dokonce ani konektor sériového portu. Když však program s s3 (ttyp3) naloží, jako by to byl sériový port, obsah toho, co se ze souboru přečte nebo zapíše do něho, se objeví v druhém členu dvojice, tedy m3 (ptyp3), jejž k zápisu a čtení používá jiný program. Dva progra-my tak spolu mohou touto metodou komunikovat a jeden program se domnívá, že hovoří se séri-ovým portem. Tento způsob si můžeme představit jako „rouru“ mezi m3 a s3.

Každý program, který je určen pro sériový port s3 (ttyp3), na něj bude psát. Ovšem druhý pro-gram, který komunikuje s m3 (ptyp3), musí být takto napsán. S pseudoterminály se musejí potý-kat hlavně programátoři, uživatelé se o ně nemusí starat.

Uveďme si příklad: Když se někdo k vašemu počítači připojí po síti přes telnet (předpokládejme, že jste telnetový server), část programu telnet zpracovávající toto spojení na vašem počítači může být napojena na pseudoterminál m2 (ptyp2). Program getty by měl běžet na odpovídajícím s2 (ttyp2). Getty si myslí, že komunikuje s terminálem. Když telnet dostane znak od vzdáleného klienta, jde přes m2 k s2 ke getty, který pak vrátí „login:“ nasměrovaný na s2, m2, program tel-net na serveru a poté po síti zpět ke klientovi. V tomto případě přihlašovací program a servero-vý telnet spolu komunikují prostřednictvím „pseudoterminálu“. Poznamenejme ještě, že na kli-entském počítači se nepoužívá pseudoterminál, pouze telnet. Server pochopitelně přidělí pseu-doterminál všem klientům.

V systému X Window používá pseudoterminály terminálový emulační program xterm (nebo rxvt). Používají je i radioamatérské programy pod Linuxem. S podporou určitého aplikačního soft-waru je možné mít 2 nebo více pseudoterminálů, jež jsou připojeny k jednomu fyzickému sério-vému portu.

V pseudoterminálových dvojicích, např. m3 (resp. ptyp3) a s3 (resp. ttyp3), je hlavním neboli řídicím terminálem m... (pty...) a s... (tty...) je terminál podřízený. Označení vychází ze sou-borového systému zařízení, kde „m“ je od slova „master“ a „s“ od slova „slave“. Pro lepší zapa-matování si můžeme říct „s“ jako „sériový“ (sériový port). V původní notaci tty. (...) odpovídá sériovému portu ttyS (původně pouze tty).

Zvýšení počtu pseudoterminálů ještě před souborovým systémem se provádělo pomocí rozšířené notace. Počet ttyp je jen 16: ttyp0-ttypf (f je hexadecimální číslice). Aby vznikl větší počet dvo-jic, písmeno p bylo nahrazeno dalšími písmeny, např. q, r, s atd. Pseudoterminálovými dvojicemi pak byly např. ttys8, ptys8 apod. Později byla přidávána další písmena. Se zavedením soubo-rového systému můžeme například pro 58. řídicí pty místo /dev/ttys9 použít /dev/pty/m57. Často se stává, že někdo místo ttyS2 (což je skutečný sériový port) uvede chybně pseudosério-vý port ttys2.

Hlavní a podřízený port jsou skutečně stejné „porty“, avšak podřízený port používají aplikační pro-gramy, zatímco hlavní port používají síťové (nebo podobné) programy, jež posílají data na podří-zený port nebo je z něho čtou. Program, který používá podřízený port, může běžet „tak jak je“, neboť se domnívá, že komunikuje se sériovým portem.
Unix98 (dostupný v Linuxu) nepracuje podle schématu shora, nýbrž používá místo toho „pty mas-ter“, což může být např. /dev/ptm3. Jeho podřízený terminál /dev/pts/3 je vytvořen automatic-ky, na požádání poskytne pty. Má se za to, že adresář /dev/pts je souborový systém typu devpts a je v seznamu připojených souborových systémů. I když „soubor“ /dev/pts/3 vypadá jako položka souborového systému zařízení, jde o zcela jiný typ souborového systému.

I když ostatní linuxové systémy mají manuálovou stránku pseudoterminálů (může se jmenovat „pty“), Linux ji pro běžného uživatele nemá. Má ji však pro programátory (openpty nebo forkpty) a předpokládá, že pseudoterminály už máte zvládnuté. Existuje jak linuxový modul pty, tak i sou-bor /usr/include/pty.h.

Řídicí terminál /dev/tty

/dev/tty je řídicím terminálem (existuje-li) stávajícího procesu. Příkazem ps -a zjistíte připojení tty ke konkrétnímu procesu. Nahlédněte do sloupce „tty“. Ve stávajícím shellovském procesu používáte terminál /dev/tty. Zjistíte to, když za promptem zadáte „tty“ (viz manuálové stránky tty(1)). Terminálech /dev/tty je něco jako odkaz na skutečné jméno terminálového zařízení s dal-šími funkcemi pro programátory v jazyce C: viz manuálové stránky tty(4).

/dev/ttyIN „terminály“

N zastupuje celé číslo. Jednou z možností použití v Linuxu je prostřednictvím balíku ovladačů ISDN: isdn4linux; ttyIN se podobá ttySN, avšak emuluje modem a lze mu zadávat příkazy modemu.

Konzoly: ttyN nebo vc/N

Monitor PC obvykle nazýváme konzola a je s ní spojeno několik speciálních souborů zařízení: vc/0 (tty0), vc/1 (tty1), vc/2 (tty2) atd. Po přihlášení jste na vc/1. Na vc/2 (na téže obra-zovce) přejdete současným stisknutím dvou kláves: levý Alt+F2, na vc/3 pomocí levý Alt+F3 atd. Tyto vc/1, vc/2, vc/3 atd. nazýváme „virtuální terminály“; vc/0 (tty0) je pouze alias pro stávají-cí virtuální terminál, na nějž jsou posílány zprávy ze systému. Systémové zprávy tedy budou vidět na konzole (monitoru) bez ohledu na to, který virtuální terminál je vypisuje.

Můžete se přihlásit na několika různých virtuálních terminálech a mít tak současně otevřených několik sezení. Na /dev/vc/0, na nějž se někdy odkazuje /dev/console, může zapisovat pouze systém a uživatel root. Další informace o konzole viz kapitola „Konzola v Linuxu“.

Vytvoření zařízení pomocí příkazu „mknod“

Adresář /dev obsahuje speciální soubory zařízení. Potřebujete-li pracovat se souborem, který v tomto adresáři není, můžete se jej pokusit vytvořit příkazem mknod. Informace o tom, jakým způsobem je nutno postupovat v případě sériových portů, naleznete v manuálových stránkách na ttys(4). Chcete-li zadat příkaz mknod, musíte nejdříve znát hlavní a vedlejší číslo zařízení. Může-te si je odvodit z výstupu příkazu ls -l v adresáři /dev, který vypíše hlavní a vedlejší čísla všech existujících speciálních souborů.

Podrobněji o tom, jak terminály pracují

Ke čtení této kapitoly jsou nutné alespoň základní znalosti o činnosti terminálů, které naleznete v „Úvodu“.

Něco o paměti

Obrazovka terminálu se z obrazu uloženého v paměti terminálu obnovuje asi 60x za sekundu. Na PC je obraz monitoru uložen na videokartě v počítači, zatímco ekvivalent videokarty terminálu je uvnitř terminálu. Textový terminál nepotřebuje velký objem paměti. V paměti nejsou uloženy všechny body (pixely) obrazu, což by vyžadovalo uložení zhruba čtvrt milionu bodů, nýbrž používá se mnohem úspornější metoda.

Obrazovka popsaná textem je v paměti terminálu uložena v bajtech ASCII, jeden bajt odpovídá jed-nomu znaku na obrazovce. Celá obrazovka zabere pouze kolem 2 k takovýchto ASCII bajtů. Tisk-nutelných znaků je téměř 100, přičemž terminál musí znát jejich bitové mapy (tvary). Když bitová mapa jednoho znaku musí být uložena řekněme v 15 bajtech, pro uložení bitových map všech znaků ASCII (neboli fontu) je potřeba pouze kolem 1,5 k paměti. ASCII text a paměť s fonty se čtou z paměti tak často, aby je bylo možno 60x za sekundu přenášet na obrazovku. Jde vlastně o urči-tý způsob sdílení paměti, neboť například pro všechna písmena „e“ na obrazovce existuje pouze jedna bitová mapa. Nízké požadavky na množství paměti znamenaly počátkem 80. let, kdy cena paměti byla několik tisíckrát vyšší než dnes (pár dolarů za kilobajt), i nízké ceny monitorů.

První terminály

Nejstarší terminály se podobaly spíše psacím strojům řízeným na dálku. Uměly pouze „zobrazit“ (vytisknout na papír) posloupnost znaků posílaných z počítače. První modely se nazývaly dálno-pisy (viz dále) a zkratka „tty“ vlastně vznikla ze slova „teletype“. Uměly posun papíru o řádek (line feed) a návrat vozíku (carriage return) podobně jako psací stroj a uměly také „cinknout“, když dostaly znak „zvonek“. Tyto terminály měly velmi málo funkcí, a byly proto nazývány „hloupé“. Tento typ „hloupého“ terminálového rozhraní se používá dodnes, když počítač neví, s jakým typem terminálu komunikuje.

Řídicí sekvence a kódy (intro)

Terminály mají mnoho funkcí, z nichž některé jsou aktivované a některé vyžadují aktivaci anebo změnu příkazem z počítače. K tomu, aby bylo možno z počítače využít všechny funkce, jsou urče-ny speciální kódy k ovládání terminálů. Existují dva hlavní typu těchto kódů: řídicí sekvence a řídi-cí kódy (resp. řídicí znaky). Řídicích sekvencí je mnohem víc než kódů.

Řídicí kódy

Řídicí kódy (lépe řečeno řídicí znaky) jsou bajty na prvních 32 místech abecedy ASCII. Jsou to: návrat vozíku (carriage-return, kurzor se posune zcela vlevo), nový řádek (line-feed, kurzor přejde
o řádek níž), posun o jeden znak zpět (backspace), znak escape (ESC), tabelátor (TAB) a zvonek (BELL). Tyto znaky se obvykle na obrazovce nezobrazují. Existuje ovšem příkaz, kterým lze moni-toru zadat, aby je zobrazoval. Příkaz se obvykle jmenuje „zobrazení řídicích znaků“ (display con-trol) nebo „monitor“ anebo nějak podobně. Když tento příkaz zadáte, na displeji vznikne kvůli řídi-cím posloupnostem zmatek, protože všechny začínají řídicím znakem ESC a po zadání tohoto pří-kazu se přestanou provádět. Slova, která by se měla nacházet na horním nebo dolním okraji obra-zovky, budou jinde. Řídicí posloupnost se sice zobrazí, ale nepřesune kurzor tam, kam má.

Řídicí posloupnosti

Vzhledem k tomu, že řídicích kódů není dost na to, aby mohly zajistit všechny funkce (a z urči-tých důvodů se některé nepoužívají), je nutno používat řídicí posloupnosti. Skládají se z řídicího znaku „escape“ (ESC), za nímž následuje posloupnost obyčejných znaků. Když terminál obdrží znak ESC, zkoumá další znaky, aby je mohl příslušným způsobem interpretovat, tedy provést požadovaný příkaz. Jakmile rozpozná platnou posloupnost, další znaky už pouze zobrazuje (pokud to nejsou řídicí znaky nebo další řídicí posloupnost). Některé řídicí posloupnosti mohou mít parametry (resp. argumenty), např. souřadnice místa na obrazovce, kam má být přesunut kurzor. Parametry jsou součástí řídicí posloupnosti. Seznam řídicích posloupností některých terminálů naleznete na Internetu, avšak není úplný.

Seznam řídicích posloupností konkrétního terminálu by měl být v „programátorském manuálu“ daného terminálu. S výjimkou velmi starých terminálů mohou tyto seznamy obsahovat několik sto-vek takových posloupností. Pokud příslušný manuál nemáte, hledají se tyto posloupnosti velice obtížně – na Internetu jsou jen některé. Některé jsou i v tomto návodu. Hledání jedné posloupnosti na Internetu (např. ESC[5m) může obnášet prohledávání velice dlouhého seznamu.

Jinou možností, jak některé z nich určit, je najít si příslušnou položku v terminfo (resp. v termcap) a rozluštit ji. Podrobnosti viz kapitola „Terminfo a termcap“, případně „Dokumentace k termin-fo/termcap“. Seznamy v terminfo, resp. termcap, bohužel často nejsou úplné, avšak nejdůležitější řídicí posloupnosti by snad měly obsahovat.

Atributy zobrazení & kouzelné cookies

Terminály mají řadu různých metod, jak vytvářejí vlastnosti znaků, např. tučné, inverzní, podtržené atd. Uživatel se o to nemusí starat, ledaže by vznikly nějaké problémy se starými terminály; na novějších terminálech je někdy nutno změnit určitá nastavení v menu „set-up“.

Metoda kouzelných cookie už poněkud vyšla z módy, je nejjednodušší (a nejhorší) metodou defi-nování atributů: jeden bajt tvoří začátek atributu a druhý jeho konec. Například cookie bajt „začá-tek podtržení“ je umístěn před prvním slovem, které má být podtrženo. Tyto řídicí bajty jsou ulo-ženy do paměti stránky obrazovky stejně jako znakové bajty, které mají být vypsány jako znaky. Může však dojít ke změně počtu znaků na řádek, neboť netisknutelné cookie znaky jsou smíchá-ny s tisknutelnými. Někdy jsou s tím problémy.

Lepší metodou, která však má větší spotřebu paměti, je přiřazení atributového bajtu (nebo půlbajtu) každému vypisovanému znaku. Tuto metodu používají při výpisu textu videokarty monitorů PC.

Další vlastnosti některých terminálů

Barva

I když běžný monochromatický monitor není barevný, displej může mít jinou barvu než bílou, např. zelenou nebo žlutou. Všechny terminály mají černou (paprsek elektronů je vypnutý = nulo-vý jas). Zatímco skutečný barevný terminál může měnit barvu textu a pozadí, monochromatický terminál pouze mění jas pevně dané barvy.

Změna jasu apod. však poskytuje určité možnosti. Například černobílý monochromatický terminál může mít díky změnám jasu bílou, šedou a černou. Některá slova mohou být černá na světle šedém pozadí, jiná výrazně černá na bílém pozadí. Dále mohou být bílá písmena na černém poza-dí, mohou být podtržená a mohou blikat.

Princip vytváření barev na terminálech je stejný jako na počítačových monitorech a na televizních obrazovkách. Povrch obrazovky je složen z bodů o třech barvách, každá barva je řízena jedním paprskem elektronů. Monochromatická obrazovka má díky konstrukci lepší rozlišení, neboť nezá-visí na pevných bodech pokrývajících obrazovku. Textové terminály používají barvy pouze k roz-lišení textů a tato výhoda nestojí vždy za horší rozlišení. Výhodou monochromatického monitoru je pochopitelně i cena.

Vícenásobná sezení

Pro dvojí sezení má terminál dva rovnocenné sériové porty. Každý port je připojen k sériovému portu jiného počítače. Lze se tedy připojit k dvěma různým počítačům, přičemž ke každému z obou sezení přísluší jedno okno rozděleného displeje. Anebo mohou obě sezení pracovat s celou obrazovkou a přepínat sezení je možné „horkou“ klávesou (nebo jakkoli jinak). Je také možné se připojit ke dvěma různým sériovým portům na jednom počítači a přihlásit se dvakrát (podobně jako „virtuálními terminály“ na konzole). Program screen zajistí, že na běžném termi-nálu jednoho počítače mohou běžet dvě nebo i více „sezení“.

Port tiskárny/pomocný port

Mnohé terminály mají na zadní straně konektor pro tento port. Může být označen „Aux“ nebo „Prin-ter“ apod. Některé porty tiskáren jsou určeny pro paralelní tiskárny, jiné pro sériové. Je-li tiskárna připojena k portu tiskárny nebo k pomocnému portu, po stisknutí určité klávesy se vytiskne obra-zovka. Lze také posílat na tiskárnu vše, co se objeví na obrazovce. V případě pomocného portu se lze připojit i k jinému počítači a vytvořit téměř dvojí sezení jako v předchozím případě. Videopa-měť v terminálu však nedokáže obnovovat obě sezení současně, je tedy možné, že ji bude nutno po přepnutí na druhé sezení obnovit „ručně“. Takový terminál pravděpodobně nebude mít horkou klávesu, pro tento účel lze však naprogramovat kteroukoli jinou klávesu. Pro řízení takového portu existuje celá řada nejrůznějších řídicích sekvencí, viz kapitola „Escape sekvence tiskáren“.

Existuje program vtprint, který je určen k odesílání tiskových úloh (pouze textových) k tisku na terminál, resp. na tiskárnu, která je k němu připojena. Domovskou stránku tohoto programu nalez-nete na adrese *http://garrett.damore.org/~garrett/software/vtprint/*. Je také součástí linuxové distri-buce Debian. Něco podobného dělá i program xprt (rovněž Debian), avšak pouze pro terminá-ly X Window??

Používání portu pro tiskárnu k tisku může být výhodné například tehdy, když na počítači nemá-te zvláštní port na připojení

tiskárny nebo pokladny. Když počítač posílá něco na terminál a ono to místo toho jde na tiskárnu, říkáme tomu „transparentní tiskový režim“. Pokud byste chtěli zajis-tit, aby tiskárna posílala bajty zpět na PC (tedy opačným směrem), nazýváme tento způsob (dle terminologie Wyse) „dvousměrný režim“. „Pomocný tiskový režim“ firmy Wyse je pouze transpa-rentní tiskový režim, při němž obrazovka terminálu monitoruje tisk.

Stránky

Celá řada terminálů může ukládat do videopaměti několik stránek. Někdy je velikost stránky stej-ná jako velikost obrazovky, někdy však je větší, takže k nezobrazeným částem stránky přejdete rolováním. Když tedy hledíte na obrazovku, skrytý text zobrazené stránky může být nahoře i dole. Navíc, je-li stránek několik, může být skrytý text součástí dalších stránek. Stránkování lze také pou-žívat na terminálech, které podporují dvojí sezení. Každé sezení může mít své vlastní stránky a lze se mezi nimi přepínat.

Dokonce i když máte jednostránkový terminál a velikost stránky je shodná s tím, co je na obra-zovce, uvidíte i další stránky souboru (apod.), které počítač posílá na terminál. Jednou z výhod přídavných stránek uložených v paměti terminálu je možnost okamžitého přechodu mezi strán-kami, aniž byste museli čekat na jejich vyslání z počítače.

Vícestránkové terminály podporuje program ncurses. Pro podporu vícestránkových terminálů existuje také komerční program „multiscreen“, pravděpodobně však nikoli v Linuxu?? Možnost práce s více stránkami je také v SCO, metoda však spíše připomíná virtuální terminály na konzole. Dále existuje linuxový program „screen“, který stránkuje výstupy, avšak stránky jsou uloženy v paměti počítače a každý program má na obrazovce k dispozici jen jedno okno.

Znakové množiny

Znakovou množinu obvykle představuje seznam (resp. tabulka nebo diagram) znaků a bajtovýchkódů. Rozsah kódů je od 0 do 255 (hexadecimálně 00 až FF). V systému MS-DOS se tabulky seznakovými množinami nazývají „kódové stránky“. S jejich obsahem byste se měli seznámit, pokudjste tak ještě neučinili. Naleznete je v dokumentaci k tiskárnám, terminálům a také pochopitelněna Internetu.

Řada množin obsahuje písmena z cizích jazyků. Mohou také obsahovat speciální znaky, které se

používají ke kreslení, a jiné.Klasickou znakovou množinou pro kódování angličtiny na terminálech je ASCII. Tento kód jesedmibitový, bude však fungovat i na terminálech, které jsou nastaveny na osmibitové kódování.V osmibitovém režimu je nejvyšší významový bit vždy nula. Jiné znakové množiny obvykle vyu-žívají ke kódování všech 8 bitů (s výjimkou velmi starých terminálů, které pracují pouze v ASCII).První polovina většiny znakových množin obsahuje 128 ASCII znaků, druhá polovina (nejvyššívýznamový bit je roven 1) pak může obsahovat širokou škálu nejrůznějších znaků. Znakové mno-žiny jsou definovány standardem ISO. Speciální znaky lze stáhnout z Internetu a nastavit do pamě-ti terminálu. Terminály mají většinou mnoho vestavěných znakových množin, avšak nikdy ne tu,kterou potřebujete.

Zde jsou některé běžně používané 8bitové znakové množiny. CP znamená Code Page (kódovástránka) firmy IBM: CP-437 (DOS ECS), ISO-8859-1 (Latin-1), CP-850 (Multilingual Latin 1 – nikolita stejná jako ISO Latin-1), CP-1252 (WinLatin1 = MS-ANSI). MS Windows používá CP-1252 (WinLa-tin1), zatímco na Internetu se často používá Latin-1. K Latin-1 se používá navíc několik množinISO-8859-, konkrétně řečtina (-7), arabština (-6), východoevropské jazyky (-2) a náhrada za Latin--1 (-15), která se nazývá Latin-9. Existuje i mnoho dalších. Například pro ruštinu se častěji použí-vá KOI8-R než IS0-8859-5. Velmi obsáhlou znakovou množinou je Unicode, v níž je každý znakzakódován ve dvou bajtech.

Další informace o znakových množinách naleznete v následujících zdrojích:

Manuálové stránky: charsets, iso_8859-l nebo latin1 (zahrnuje řadu 8859), ascii.
Návody k různým jazykům (často psané právě v oněch jazycích).
ISO-8859 Alphabet Soup (*http://czyborra.com/charsets/iso8859.html*). Více než jen iso8859. Rozsáhlé.
A tutorial on character code issues (*http://www.cs.tut.fi/~jkorpela/chars.html*). Psáno sro-zumitelně.
Languages, Countries and Character Sets (*http://www.w3.org/International/O-charset-lang.html*).
Languages of the World by Computers ... (*http://www.osk.3web.ne.jp/logos/*).
Links re Internationalization (*http://linux.monnet.ru/books/locale/locale_i.html*). Dlouhý seznam odkazů (rusky, avšak většina slov je anglického původu).
... International Character Sets (ftp://kermit.columbia.edu/kermit/e/isok7.txt).

Když znáte jméno znakové množiny (nebo její alfanumerické označení), další informace o ní snadno naleznete na Internetu.

Grafika (Kreslení čar apod.)

Pro kreslení čtverečků atd. existují speciální znaky. Dále existuje mnoho neASCII symbolů, např. puntíky. Mohou být buď součástí 8bitových množin (např. WinLatin1 = CP-1252) nebo existují jako samostatný font (terminály vt100). Někdy jsou obsaženy v terminfo, avšak vidíte-li v tabulce slou-pec, který by měl být řádkem, může to znamenat, že v terminfo nejsou implementovány.

Nefunguje-li vám grafika, jak byste očekávali, měli byste vědět alespoň toto: vt-100 ANSI je implicitní grafická množina, jinak musí být v terminfo definován řetězec acs, který definuje mapu mezi kódy grafických znaků vt-100 a kódy, které používá terminál. Když tedy terminál nemá grafiku vt-100, může ji vytvářet pomocí jiné znakové množiny. Terminfo se správným obsahem to činí automaticky.

Je-li nutno vyměnit znakové množiny, musí být definovány tyto proměnné terminfo: enacs, rmacs a smacs. Poznamenejme, že acs = Alternate Character Set (alternativní znaková množina). I když horní polovina normální znakové množiny obsahuje grafické znaky, může se mít za to, že je nutno vyměnit samostatnou 7bitovou množinu.

Národní znaky (historie)

V roce 1960 byl vynalezen sedmibitový kód ASCII, který mapoval sedmibitové bajty na anglická písmena, číslice, diakritická znaménka atd. Ostatní země jej přijaly, avšak některé znaky v něm postrádaly. Co měly dělat? Někdo se rozhodl některé symboly (např. ^, }) odstranit a nahradit je národními písmeny (které třeba mají nad sebou tečku apod.). Jinými slovy, ASCII znaky byly nahrazeny „národními znaky".

Byly s tím obrovské problémy, neboť náhrady prováděly většinou firmy, které pak prodávaly ter-minály bez jakýchkoli standardů. Dalším problémem bylo, že odstraněné znaky chyběly. Tyto pro-blémy byly řešeny v 80. a 90. letech tak, že se v bajtu využilo všech 8 bitů a znakové množiny mohly obsahovat mnohem víc znaků.

Mnohé západoevropské jazyky potřebovaly přidat jen několik znaků, které nebyly v ASCII.
V sedmibitovém kódu si vypůjčily málo používané symboly: @ [\ ] ^ ` { \ } ~. Také někdy použily znaky $ a #, čímž naopak vznikla omezení, neboť místo nich se používalynové národní znaky, které nebyly součástí ASCII. Pak, když 8bitové znaky nahradily 7bitové,mohly znakové množiny obsahovat jak všechny znaky ASCII, tak i národní znaky různých jazyků.Byl také vytvořen Unicode, který nahradil 8bitové kódy týmž kódovým schématem, aby pokrylvšechny jazyky (anebo téměř všechny).

Používání (7bitových) národních znaků bylo kodifikováno v ISO-646 (r. 1972 a další). Tato norma pouze sděluje, že shora zmíněné znaky si lze vypůjčit, avšak už neříká, kterými národními mají být nahrazeny. Některé země standardizovaly náhrady tak, že je zaregistrovaly v ECMA.

Národní znaky jsou podporovány celou řadou terminálů, avšak pokud tuto podporu nepotřebujete pro čtení starších souborů, asi ji nebudete implementovat. Velmi staré terminály možná podporují pouze národní znaky těch zemí, pro něž byly určeny, zatímco novější terminály už nabízejí výběr. Modemové terminály jsou osmibitové a nemusí mít národní znaky. Národní znaky existují pro tyto jazyky/země: britská angličtina, kubánská španělština, holandština, finština, francouzština, kanadská francouzština, němčina, hebrejština, maďarština, italština, norština/dánština, portugalština, španělština, švédština, švýcarská němčina.

Zde je tabulka některých znakových množin převzatá z dokumentů Kermit a Unisys:

švédština dánština

ASCII němčina finština norština francouzština

@ zavináč rozděl. ----------a-grave [levá hranatá A-přehlas. A-přehlas. AE-dvojpísm. stupeň / lomítko O-přehlas. O-přehlas. O-přeškrt. c-ocásek] pravá hranatá U-přehlas. A-circle A-kroužek rozděl. ^ stříška ------U-přehlas. ----------` apostrof ------e-acute ----------{ levá složená a-přehlas. a-přehlas. ae-dvojpísm. e-acute | svislá čára o-přehlas. o-přehlas. o-kroužek u-grave } pravá složená u-přehlas. a-kroužek a-kroužek e-grave ~ tilda ostré s u-přehlas. --------i-přehlas.

ASCII italština španělština

@ zavináč rozděl. rozděl. [levá hranatá degree obr. vykřičník / lomítko #-pound N-tilda] pravá hranatá e-acute obr. otazník ^ stříška ----------` apostrof u-grave -------{ levá složená a-grave stupeň | svislá čára o-grave n-tilda } pravá složená e-grave --------~ tilda i-grave --------

Fonty

Většina terminálů vyrobených po roce 1985 si fonty umí načíst do paměti (tzv. soft-fonty). To zna-mená, že umí psát prakticky libovolnými fonty, pokud existují jako soft-fonty. Jestliže vám žádné nevyhovují, můžete si vytvořit fonty vlastní. K tomuto účelu existuje editor jménem BitFontEdi (napsal jej autor tohoto dokumentu) a v Evropě je (od roku 1998) k dispozici na *http://www.funet.fi/pub/culture/russian/comp/cyril-term/*, v Severní Americe na: *http:// ibiblio.unc.edu/pub/Linux/utils/terminal/*. Mapování různých fontů na klávesnici a obrazovku naleznete v části „Mapování znaků: mapchan".

Klávesnice & speciální klávesy

Klávesnice terminálu mívají řadu kláves, které ani na klávesnici PC nenajdete. Některé terminály (je jich málo, pokud vůbec) mají všechny klávesy zde uvedené, některé mají i takové, které v tomto přehledu nejsou. Některé mají řadu speciálních kláves, například terminály používané jako registrační pokladny. Klávesy mívají i jiné významy než ty, které jsou zde uvedeny, zejména ve spojení s jinými klávesami (například shift a control).

Klávesa BREAK posílá na počítač velmi dlouhou 0 (mezera = +12 V) v trvání od 300 do 700 milisekund. Má-li stty nastaven brkint, počítač ji interpretuje jako přerušovací klávesu, jinak ji ignoruje.
Klávesa NO SCROLL zruší rolování obrazovky podobně jako ^S. Když klávesu povolíte, rolování pokračuje. Používá k tomu signály pro řízení toku.
Když klávesu REPEAT přidržíte stisknutou současně s jinou klávesou, posílá výstup opa-kovaně, a to i tehdy, když není nastavena volba „opakování".
Klávesa LINE FEED pošle do počítače znak ^J. Používá se zřídka.
Pomocí klávesy SET-UP můžete vyvolat menu pro nastavení terminálu. Někdy je úmyslně zablokována podložkou tak, aby nešla stisknout. Někdy je zase nutno současně stisknout jinou klávesu, viz kapitola „Konfigurační režim".
Klávesa LOCAL odpojí terminál od počítače. V lokálním režimu jde znak po stisknutí klá-vesy přímo na obrazovku, což je vhodné například na testování.
Klávesa RETURN odpovídá klávese Enter na PC. Obvykle odešle na počítač znak „návrat vozíku" (carriage return) a ovladač jej převede na znak „nový řádek". Na některých termi-nálech může být nastaven tak, že posílá jiný znak.
F1, F2, ... nebo PF1, PF2, ... jsou funkční klávesy, které je možno naprogramovat tak, aby odeslaly posloupnost bajtů (znaků), viz kapitola „Funkční klávesy".

Myš

Některé textové terminály podporují myš. Když na ni klepnete, na počítač se odešle řídicí sekven-ce, která sdělí, kde je myš umístěna. O myších na terminálech VT viz *http://www.cs.utk.edu/ ~shuford/terminal/dec_vt_mouse.html.* Tyto řídicí kódy pro myši se nazývají „DEC posloupnosti umístěni". Model ANSI-G z terminálové řady FALCO Infinity ji podporuje. Existuje i linuxová apli-kace, která by ji podporovala??

Emulace terminálů (včetně konzole)

Úvod do emulace terminálů

Vzhledem k tomu, že PC má obrazovku a klávesnici (podobně jako terminál), ale také má velkou výpočetní kapacitu, je jednoduché využít její část k tomu, aby se osobní počítač choval jako tex-tový terminál. To je jeden typ emulace terminálů. Jiný typ emulace terminálu je, když skutečný terminál emuluje jiný model terminálu. Z nabídky pro nastavení si vyberete určitou emulaci (v žar-gonu Wyse „charakteristiku"). Tato kapitola popisuje první typ emulace: emulaci terminálu na PC.

Při emulaci použijete jeden sériový port počítače pro připojení emulovaného terminálu k druhé-mu počítači. Buď propojíte oba sériové porty kabelem nebo přes modem. Emulace může poskyt-nout více funkcí než terminál, neboť PC, který provádí emulaci, může současně pracovat i na jiných úlohách. Mohou na něm běžet kermit nebo zmodem a po sériové lince přenášet na počí-tač, k němuž jste připojeni, soubory (anebo i po telefonní lince modemem). Musí být pouze zajiš-těno, aby emulace běžela na jedné z virtuálních konzoly PC a ostatní virtuální konzoly tvořily roz-hraní pro zadávání příkazových řádků.

Pod operačním systémem MS Windows je k dispozici dostatek emulačního softwaru, viz „Jak vytvořit z nelinuxového PC terminál". Tímto způsobem můžete propojit počítač s operačním systémem MS Windows s počítačem vybaveným Linuxem (jako textovým terminálem). Většina volně dostupného softwaru v Linuxu umí emulovat pouze VT100, VT102, resp. VT100/ANSI. Víte-li o jiných, dejte mi, prosím, vědět. Vzhledem k tomu, že osobní počítače mají barevné monitory, zatímco terminály VT100 a VT102 jsou monochromatické, emulace jim umožňuje navíc práci s bar-vami (včetně výběru barev). Někdy emulace není úplně stoprocentní, avšak nejsou s tím zásadní problémy. Emulace terminálů na počítačích Mac viz Mininávod k Mac terminálům.

K emulaci nepoužívejte proměnnou TERM

Občas někoho napadne scestná myšlenka, že by mohl vytvořit emulátor na linuxové konzole (monitoru) tak, že do vnější proměnné TERM nastaví typ terminálu, který chce emulovat. To nefun-guje. Hodnota TERM pouze říká aplikaci, který právě používá terminál, a tudíž se aplikace nemu-sí ptát interaktivně. Pracujete-li s monitorem PC (jako s rozhraním pro příkazové řádky), typ ter-minálu je „Linux" a nemůžete to změnit. TERM tedy musí být nastaven na „Linux".

Nastavíte-li tuto proměnnou jinak, uvádíte aplikační program v omyl, což povede k chybné inter-pretaci některých řídicích sekvencí z konzoly a poté i ke zhroucení rozhraní. Vzhledem k tomu, že se linuxová konzola chová téměř jako terminál vt100, může při nastavení TERM na vt100 (nebo podobný) konzola fungovat po většinu doby téměř bezchybně a k chybě může dojít třeba při edi-tování apod.

Komunikační (vytáčecí) programy

Vytáčecí programy, které na Internetu uskutečňují dvoubodové spojení, obvykle neobsahují emu-laci terminálů. K těm nečetným, které ji obsahují, patří například minicom nebo seyon. Jejich pro-střednictvím můžete například vytočit veřejnou knihovnu a nahlížet do seznamů a rejstříků (anebo dokonce číst články v časopisech). Lze jimi také testovat modemy. Program seyon lze použít pouze v systému X Window a umí emulovat terminály Tektronix 4014.

Komunikační program Kermit neprovádí emulaci terminálů, je pouze polotransparentní rourou mezi libovolným terminálem, na němž pracujete, a vzdáleným sezením, k němuž jste připojeni. Používáte-li tedy kermit na linuxovém počítači, typ terminálu bude „Linux“. Máte-li k PC připojen Wyse60 a provozujete-li na něm kermit, vzdálenému počítači se budete jevit jako Wyse60 (s nímž třeba neumí komunikovat). Minicom emuluje VT102, a používáte-li terminál Wyse60, řídicí sekven-ce terminálu VT102 přicházející ze sériového portu vzdáleného počítače budou před odesláním na druhý sériový port terminálu Wyse60 převedeny na řídicí sekvence VT102, což kermit neumí.

Emulátory *telix* a *procomm* pod systémem DOS pracují stejně. Emulovanými terminály jsou obvyk-le staré VT100, VT102 nebo ANSI (podobný VT100).

Emulace pod X Window

Xterm (resp. uxterm, který na rozdíl od xterm podporuje unicode) může běžet pod X Window. Umějí emulovat VT102, VT220 a Tektronix 4014. Existují také různé emulace xterm (i když žádný terminál jménem „xterm“ neexistuje). Chcete-li pixmaps, avšak nepotřebujete emulaci Tektronix 4014 (terminál s vektorovou grafikou; viz „Grafické terminály“), můžete zvolit eterm. Předchůdci programu eterm jsou rxvt a xvt. Jedním z možných způsobů, jak lze změnit velikost fontu v xter-mu, je pravé klepnutí myší se současně stisknutou klávesou Ctrl.

V případě jiných abeced než latinka umí kterm emulovat terminál Kanji (nebo i jiné nelatinské abecedy); xcin je určen pro čínštinu. Existuje také emulace 9term. Zdá se, že v tomto případě jde
o více než jen emulátor a že má vestavěný editor a posuvné lišty. Byl navržen pro Plan 9, což je unixový operační systém od AT&T.

Lepší jsou skutečné terminály

Pokud nepoužíváte X Window s velkým displejem, skutečný terminál je lepší než emulovaný. Má obvykle lepší rozlišení textu a neobtěžuje hlukem diskové mechaniky.

Test emulace terminálu

K terminálové řadě VT existuje testovací program vttest, který pomáhá určit, zda se terminál chová korektně jako vt53, vt100, vt102, vt220, vt320, vt420 atd. Není k němu dokumentace, má však nabídku a snadno se ovládá. Přeložíte jej pomocí konfiguračního skriptu a zadáte make. Lze jej stáhnout z *http://ibiblio.unc.edu/pub/Linux/utils/console/*.

Konzola Linuxu

Konzola systému Linux je obvykle monitor počítače v textovém režimu. Emuluje terminál typu „Linux“ a řídicí sekvence, kterou používá, je uvedena v manuálových stránkách: console_codes. Neexistuje způsob (pokud nechcete trávit týdny přepisováním jádra), jak ji přinutit k emulaci něče-ho jiného. Když nastavíme vnější proměnnou TERM na jakýkoli jiný typ terminálu než „Linux“, neznamená to, že se provede jeho emulace. Pouze může dojít k poškození rozhraní, neboť jste nesprávně deklarovali (proměnnou TERM), že váš „terminál“ se od uvedeného typu liší. Viz kapi-tola „Pro emulaci nepoužívejte vnější proměnnou“.

V některých případech je konzola linuxového počítače textový terminál. Linux je možno přeložit tak, že většina hlášení, která obvykle jdou na konzolu, půjde na terminál. Viz „Jak udělat konzolu ze sériového terminálu“.

Linuxová emulace monitoru je flexibilní a oproti terminálu vt102, který měl být emulován, má řadu dalších funkcí. Umí například používat zákaznické fonty a snadno je možno přemapovat kláves-nici. Tyto funkce zajišťuje ovladač konzole (společně s ovladačem klávesnice), který pouze pra-cuje pro monitor, nikoli pro skutečný terminál, i kdyby se používal jako konzola. „Ovladač kon-zoly“ tedy je ve skutečnosti „ovladačem monitoru“. Když byl Linux ještě v plenkách, nebylo možno používat skutečný terminál jako konzolu, takže tehdy nebyl mezi „monitorem“ a „konzolou“ žádný rozdíl.

Příkazy stty pracují na monitoru-konzole stejně, jako by to byl skutečný terminál. Ovládá je stej-ný ovladač terminálů, který se používá i pro skutečné terminály. Bajty směřující na obrazovku pro-cházejí nejdříve ovladačem terminálu (tty) a pak ovladačem konzoly. Některé příkazy stty nedě-lají na monitoru nic (např. nastavování přenosové rychlosti). Přenosovou rychlost monitoru může-te nastavit libovolně (např. na pomalých 300), avšak skutečná rychlost psaní textu na obrazovku monitoru se nezmění. V souboru /etc/ioctl.save je uloženo nastavení stty pouze tehdy, když je konzola v jednouživatelském režimu (obvykle však je ve víceuživatelském režimu). Trochu je to vysvětleno v manuálové stránce init.

Mnohé příkazy jsou určeny k využívání rozšířených funkcí, které nabízí ovladač konzoly-monito-ru. Skutečné terminály, které nepoužívají ani sken kódy, ani VGA karty, bohužel tyto funkce nevy-užijí. Další informace o konzole najdete v Keyboard-and-Console-HOWTO. Také je možno nahlédnout do různých manuálových stránek o konzole (zadejte příkaz man -k console). Větši-na této dokumentace je ovšem bohužel zastaralá.

Emulační software

Emulátory občas nefungují tak, jak by měly, takže byste si nejdříve měli důkladně prověřit, co kupujete.

Vytvořte si linuxový terminál

Pokud nechcete emulovat standardní vt100 (nebo podobný terminál) nebo Wyse 60, v Linuxu není k dispozici příliš mnoho volně dostupného emulačního softwaru. Programy minicom a seyon (pouze pro X Window) emulují vt100 (a podobné). Seyon také umí emulovat terminál Tektronix 4014, viz Wyse 60 emulator (*http://gutschke.com/wy60/*).

Pro emulaci přímo napojeného terminálu lze použít minicom, a to tak, že jej pouze spustíte (před-tím je ho ovšem nutno zkonfigurovat na sériový port). Pochopitelně, když chcete skončit (a odhlá-sili jste se z druhého PC), spojení neukončíte jako vytáčené, nýbrž zadáte příkaz q (quit) bez rese-tování, protože není co resetovat (není přítomen modem). Jakmile je mincom spuštěn, automa-ticky vyšle na sériový port inicializační řetězec modemu. Avšak vzhledem k tomu, že připojení není uskutečněno modemem, řetězec se vypíše za promptem „login:“. Obsahuje-li tento řetězec převážně velká písmena, program getty (který spouští login) na protějším PC by se mohl domní-vat, že váš terminál umí psát jen velkými písmeny, a mohl by se pokusit komunikovat pouze pro-střednictvím velkých písmen. Tomu zabráníte, když zkonfigurujete minicom tak, aby posílal prázdný inicializační řetězec modemu.

Emulátor terminálů „Procom“ (z DOSu) lze používat na linuxovém počítači, když spustíte dosemu na emulování DOSu. Podrobnosti viz *http://solarflow.dyndns.org/pcplus/.* Existuje speciální linuxová distribuce: Serial Terminal Linux, která změní PC na terminál typu mini-com. Je malý (vejde se na pružný disk) a neumožní jiné použití PC (když běží). Viz *http://www.eskimo.com/~johnnyb/computers/stl/.* Umožňuje současný běh několika sezení (podob-ně jako v případě virtuálních terminálů), jedno pro každý sériový port.

TERM (jedná se o komerční program od firmy Century Software, který naleznete na adrese *http://www.ecc400.com/censoft/termunix.html*) umí emulovat Wyse60, 50; VT 220, 102, 100, 52: TV950, 925, 912; PCTERM; ANSI; IBM3101; ADM-1l; WANG 2110. Pro IBM a Wyse existuje blo-kový režim. Běží na linuxovém PC.

Vytvořte si nelinuxový terminál

Existují i emulátory, které běží na nelinuxových PC. Umožňují použití nelinuxových PC jako ter-minálů připojených k linuxovým PC. V systému DOS jsou to programy telix a procomm. Windows nabízí emulátor „HyperTerminal“ (původně ve Windows 3.x a v DOSu nazývaný jen „Terminal“).

Konkuruje mu „HyperTerminal Private Edition“ (*http://www.hilgraeve.com/htpe/index.html*), který není zdarma pro komerční organizace. Umí emulovat vt220. Terminály v systémech Windows jsou určeny pro napojení přes modem, měly by však fungovat i při přímém napojení terminálů?? Pro-gram firmy Turbosoft TTWin umí emulovat přes 80 různých terminálů pod Windows, viz *http://www.ttwin.com/* nebo *http://www.turbosoft.com.au/* (Austrálie). Viz také Reflection, *http://www.wrq.com/.*

Na počítačích Mac se emulace provádí programy firmy Carnation Software, *http://www.carnation-software.com/carnation/HT.Carn.Home.html.*

Jedním z míst, kde lze ověřit produkty pro emulaci terminálů, jsou stránky firmy Shuford, avšak zdá se, že seznam produktů je poněkud zastaralý (i když mohou stále fungovat). Skutečnost, že většina z nich běží pouze pod systémem DOS (a nikoli Windows) nasvědčuje tomu, že uvedené informace nejsou nejčerstvější. Viz *http://www.cs.utk.edu/~shuford/terminal/term_emulator_ products.txt.*

Řízení toku dat

Řízení toku dat (= handshaking = pacing) slouží k tomu, aby příliš rychlý toku bajtů nezahltil ter-minál, počítač, modem nebo jiné zařízení. K zahlcení dochází tehdy, když zařízení není schopno dostatečně rychle zpracovat data, která dostává, a proto dochází ke ztrátě bajtů a k dalším závažným chybám. Řízení toku pozastaví přísun dat, dokud (například) terminál není znovu připraven k pří-jmu dalších bajtů. Řízení toku dat posílá signál k pozastavení toku v opačném směru, než přicháze-jí bajty, jež se mají zastavit. Řízení toku musí být nastaveno jak na terminálu, tak i na počítači.

Existují dva typy řízení toku dat: hardwarové a softwarové (Xon/Xoff nebo DC1/DC3). Hardwa-rové řízení používá jednoúčelové signální vodiče, např. RTS/CTS nebo DTR/DSR, zatímco soft-warové řízení používá k řízení řídicí bajty DC1 a DC3 v běžných datových vodičích. Při hardwa-rové kontrole musí být vodiče správně zapojeny.
Tok datových bajtů v kabelu mezi dvěma sériovými porty je dvousměrný, takže musíme uvažovat o dvou různých tocích (a vodičích):

Tok bajtů z počítače na terminál.
Tok bajtů z klávesnice terminálu do počítače.

K čemu je řízení toku dat dobré?

Mohli byste se zeptat: „Proč se data neposílají tak pomalu, aby nezahltila zařízení? Pak by bylo řízení toku dat zbytečné.“ To by sice bylo možné, avšak výsledek by byl výrazně pomalejší než při řízení toku. Jednou z příčin je skutečnost, že přenosovou rychlost sériového portu nelze nasta-vit na libovolnou rychlost jako třeba 14 500, neboť je možno vybrat jen z omezeného počtu rych-lostí. Nejlepší je nastavit rychlost o trochu větší, než jakou je schopno zvládnout zařízení, a pomo-cí řízení toku ji optimalizovat.

Pokud se rozhodnete řízení toku dat nevyužívat, je nutno nastavit přenosovou rychlost tak nízkou, aby zařízení v každém případě zvládlo i nejhorší možné situace. V případě terminálu je to odesílání řídicích sekvencí ve složitých úlohách, které zaberou více času než obvykle. Pokud jde
o modem (s komprimovanými daty, nikoli však s řízením toku dat), rychlost odesílání dat z počí-tače na modem musí být tak nízká, aby ji zvládala i telefonní linka, neboť nejhorším případem je přenos nekomprimovaných dat. Pokud nepoužíváte řízení toku dat, rychlost (se zapnutou kom-primací dat) nebude větší než bez komprimace.

V nejhorších situacích mohou částečně pomoci bufery, ovšem pouze v krátkodobých situacích. Data, která nemohou být zpracována ihned, se přechodně ukládají v buferu a jsou zpracována později.

Vycpávky

Jinou možností, jak je možno zvládat „nejhorší případ“ (bez použití řízení toku dat nebo buferů), je doplňování řídicích sekvencí nulami (bajty s nulovou hodnotou). Někdy se místo nul používají znaky DEL, pokud nemají žádnou jinou funkci, viz kapitola „Znak Del“.

Na základě řídicí sekvence začne terminál provádět nějakou činnost, a než ji dokončí, po lince přichází posloupnost nul, kterou ignoruje. Říká se tomu vycpávka nulovými bajty. Než terminál dostane poslední nulový bajt, je s úlohou hotov a může začít se zpracováním dalšího příkazu. Tyto bajty se dříve nazývaly „výplňkové znaky“ a přidávají se jen proto, aby pozdržely přenos po dobu, kdy je terminál zaneprázdněn. Tento způsob byl před tím, než přišlo do obliby řízení toku dat, poměrně rozšířený. Jeho efektivita závisela na správném počtu přidaných nul, přičemž zjišťování optimální hodnoty bylo dosti otravné. Většinou se provádělo metodou pokusů a omylů, protože návody k terminálům byly v tomto ohledu nepoužitelné. Pokud řízení toku dat nepřináší očeká-vané výsledky anebo není k dispozici, vycpávky mohou být jedním z řešení. Vycpávkám je urče-na jedna volba v příkazu stty.

Zahlcení sériového portu

Mohlo by se zdát divné, jak může docházet k zahlcení sériového portu, když odesílání i přijímá-ní prostřednictvím sériového portu je nastaveno na stejnou přenosovou rychlost (v bitech za sekundu), například 19 200. Důvodem je skutečnost, že i když elektronika zajišťující příjem na séri-ovém portu zvládne zpracování příchozího toku dat, může s ním mít problémy hardware i soft-ware terminálu.

Jednou z příčin tohoto stavu je malý hardwarový bufer sériového portu. Starší sériové porty měly bufer o velikosti pouze jednoho bajtu (v mikroprocesoru UART). Pokud si tento bajt z buferu nevyzvedne instrukce základní jednotky ještě předtím, než přijde další bajt, původní bajt je ztra-cen (došlo k zahlcení). Nové mikroprocesory UART, zejména většina řady 16550, mají bufery šest-náctibajtové (avšak mohou emulovat jednobajtové) a k zahlcení dochází méně. Může být nasta-ven i tak, že dosáhne-li obsah buferu 1, 4, 8 nebo 14 bajtů, vydá přerušení a jiný mikroprocesor (většinou základní jednotka) vyzvedne obsah příchozího buferu a zpracuje ho (podobně jako jiné úlohy).

Když obsah tohoto malého hardwarového přijímacího buferu dosáhne určité hranice (u starých UART jeden bajt), vydá přerušení. Počítač přeruší práci a softwarově zjistí, co se stalo. Poté bajt (nebo několik bajtů) z tohoto buferu sériového portu vyzvedne. Přenese je do většího buferu (který také patří k sériovému portu) v hlavní paměti jádra. Po vyprázdnění (nebo částečném vyprázdnění) vysílacího buferu vydá sériový port další přerušovací signál, aby sdělil základní jed-notce, že je bufer volný a lze jej případně naplnit odesílanými daty.

Terminály také mají sériové porty a bufery, podobně jako počítač. Vzhledem k tomu, že tok dat je obvykle směrem do terminálu mnohem větší než v opačném směru (z klávesnice do počítače), k zahlcení dochází častěji v terminálu. Počítač používaný jako (emulovaný) terminál bude pocho-pitelně k zahlcení náchylnější.

Rizikové situace, kdy je zahlcení pravděpodobnější, nastávají, když: 1. Jiný proces zablokuje přerušení. 2. Téměř přetéká bufer sériového portu v hlavní paměti nebo v paměti terminálu.

Neposílat!

Když se zdá, že dochází k zahlcení přijímače příchozími daty, přijímač vyšle signál odesílateli, aby ukončil odesílání. Jde o signál řízení přenosu dat, který je posílán opačným směrem než data, jež jsou jím řízena (avšak jiným kanálem, resp. po jiném vodiči). Tímto signálem může být buď řídi-cí znak (^S = DC3 = Xoff) poslaný jako obyčejný datový bajt po datovém vodiči (signalizace v rámci pásma) nebo přechod od kladného elektrického potenciálu k zápornému na signálním vodiči dtr-to-cts (případně jiném), což je signalizace mimo přenosové pásmo. Použití Xoff se nazý-vá „softwarové řízení toku“, zatímco elektronická signalizace po určeném vodiči (v kabelu) se nazývá hardwarové řízení toku.

Uzamknutí klávesnice

Nejobvyklejším případem ukončení posílání dat je, když terminál není schopen udržet krok se znaky, které dostává, a vydá počítači signál „stop“. Jiným případem je stisknutí kláves control--S. Méně obvyklý je opačný případ, kdy počítač nestačí

rychlosti psaní a vydá terminálu signál k ukončení přenosu. Terminál v takovém případě „uzamkne“ klávesnici a obsluhu o tom infor-muje vypsaná zpráva nebo rozsvícená kontrolka. Další psaní na klávesnici je ignorováno. Jakmile se počítač vzpamatuje, klávesnici odemkne. Když se tak nestane, někde asi nastal deadlock (pro-cesy uvízly na mrtvém bodě).

K jinému typu uzamknutí klávesnice dojde, když je na terminál poslána určitá řídicí sekvence (v případě Wyse 60 pouze řídicí znak ^O). Zatímco v předchozím typu provede uzamknutí klá-vesnice sériový ovladač, v tomto případě klávesnici uzamkne hardware terminálu. Je to situace jako vystřižená z románu Hlava 22, neboť nastane-li, není možné zadat příkaz, kterým byste ji změ-nili. Možná ji lze odstranit restartem počítače (avšak svůj terminál Wyse 60 jsem dokonce musel odpojit od sítě). Bylo by také možné poslat „odemkni klávesnici“ z jiného terminálu.

Označení klávesnice jako „uzamknuté“ se někdy používá i v případě uvedeném shora, kdy počí-tač pouze přeruší posílání znaků na terminál. Klávesnice ve skutečnosti není uzamknuta a znaky z klávesnice jdou do počítače. Avšak vzhledem k tomu, že počítač nemůže odpovídat, znaky psané na klávesnici se neobjevují na obrazovce a klávesnice vypadá jako zamknutá. Je zamknuté i rolování, nikoli však klávesnice samotná.

Shrnutí

Když přijímač dožene zpoždění a může pokračovat v přijímání dat, signalizuje tuto skutečnost odesílateli. V případě softwarového řízení je signálem řídicí znak ^Q = DC1 = Xon, který je posí-lán po normální datové lince. Při hardwarovém řízení toku dat se změní potenciál na signální lince ze záporného na kladný. Pokud je terminál požádán o obnovu přenosu, klávesnice je odemknu-ta a připravena k dalšímu použití.

Hardwarové řízení toku dat (RTS/CTS atd.)

Některé starší terminály hardwarové řízení toku nemají, zatímco jiné pro ně využívají širokou nabídku různých pinů na konektoru sériového portu. Seznam pinů a jejich označení viz kapitola „Kabel pro standardní nullmodem“. Nejoblíbenějším pinem pro tyto účely je pravděpodobně pin DTR (nebo dvojice DTR a DSR).

Řízení toku dat pomocí RTS/CTS, DTR a DTR/DSR

Linuxový počítač používá řízení toku dat prostřednictvím RTS/CTS, avšak řízení pomocí DTR/DSR (které používají některé terminály) je velmi podobné. Řízení toku pomocí DTR (pouze v jednom směru a také jen u některých terminálů) je pouze součástí řízení DTR/DSR.

Řízení RTS/CTS používá piny RTS a CTS na sériovém konektoru EIA 232. RTS znamená „Request To Send“. Je-li na tomto pinu na straně příjemce kladné napětí, znamená: posílej mně data. Změna napětí na záporné (negace RTS) znamená: neposílej (zastav posílání). Jakmile je přijímač připra-ven na další vstup, změní napětí na RTS na kladnou hodnotu, což znamená žádost o obnovení posílání dat. Jak na počítači, tak i na terminále (typ obou je DTE) se signál z pinu RTS odešle na pin CTS (Clear To Send) na druhém konci kabelu. To znamená, že pin RTS na jednom konci kabe-lu je propojen s pinem CTS na druhém konci kabelu.

V případě modemu (zařízení DCE) je schéma odlišné, neboť signál přijímá pin RTS a odesílá jej pin CTS. I když to může působit zmateně, existují k tomu vážné historické důvody, které by bylo složité zde probírat.

Terminály obvykle mají řízení toku dat DTR nebo DTR/DSR. DTR je totéž co DTR/DSR s tím roz-dílem, že je pouze jednosměrné a pin DSR se nepoužívá. U DTR/DSR na terminálu je signál DTR jako signál poslaný z pinu RTS a pin DSR je jako pin CTS u nullmodemu (překřížení).

Zapojení konektoru

Některé terminály používají jenom řízení toku dat DTR. Je to pouze jednosměrné řízení, které chrání terminál před zahlcením. Nechrání počítač před rychlým psaním. Na standardním sériovém kabelu pro přenos souborů je pin DTR na straně terminálu připojen k pinu DSR na straně počíta-če. Linux však řízení DTR/DSR nepodporuje (možná je podporují některé ovladače multiportových desek). Tento problém můžete obejít tak, že jenom propojíte pin DTR na terminále s pinem CTS na počítači a nastavíte řízení toku RTS/CTS (stty crtscts). Skutečnost, že tento způsob je jediný, nebude mít vliv vůbec na nic, ledaže by počítač nestíhal zpracovat vaše rychlopsaní a provedl marný pokus o uzamknutí klávesnice, viz kapitola „Uzamykání klávesnice“. Obousměrné řízení DTR/DSR (pokud je podporuje terminál) viz text výše. Také ovšem propojte pin DSR na terminá-le s pinem RTS na počítači, abyste byli chráněni pro případ, že píšete moc rychle.

Jak se liší původní řízení toku RTS/CTS

Matoucí je skutečnost, že původní význam RTS je opakem toho, co jsme si vysvětlili výše. Původ-ní význam RTS je: *I Request To Send to you* (žádám, abych vám mohl posílat). Tento požadavek by měl být poslán z terminálu (nebo z počítače) na modem, který, pokud by se rozhodl tento požadavek přijmout, by poslal ze svého pinu CTS na pin CTS počítače potvrzení: *You are Cleared To Send to me* (můžete posílat). Poznamenejme, že narozdíl od současného obousměrného řízení toku RTS/CTS bylo původní řízení pouze jednosměrné: z počítače (nebo z terminálu) na modem.

U starších terminálů může mít RTS tento význam, a má-li terminál data na odeslání, jde do klad-ných hodnot. Použití výše je jistá

forma způsobu řízení toku dat, neboť pokud modem chce, aby počítač přerušil posílání dat, shodí CTS (připojenou k CTS na počítači) a počítač přestane posílat data.

Opačný kanál

Staré terminály, které psaly na papír, mívaly piny pro opačný kanál (možná měl číslo 19), který fungoval jako pin RTS při řízení toku RTS/CTS. Když došel papír nebo barvicí páska, na pinu se objevilo opačné napětí. Většinou stačilo připojit tento pin k pinu CTS centrálního počítače, kde býval přepínač polarity signálu.

Je hardwarové řízení toku dat prováděno hardwarem?

Je nutno říci, že hardwarové řízení toku dat je prováděno hardwarem pouze částečně, většinu provádí operační systém. Mikroprocesory UART a ostatní hardware ve skutečnosti o řízení toku dat nic nevědí. Přerušovací signál je pouze předán základní jednotce, aniž by byl hardwaru znám jeho význam. Základní jednotka přeruší zpracování a pomocí tabulky přerušení v operační paměti vyhledá program, který přerušení zpracuje a rozhodne o dalším postupu. V tomto případě přeruší výstupní tok dat.

Tok dat je ovšem zastaven už při zpracování přerušení základní jednotkou, čímž vlastně je toto zastavení provedeno rychleji, neboť není nutno čekat, až tak učiní program. Pokud by ovšem pro-gram nevydal příkaz k zastavení toku dat, po ukončení programu by se tok dat obnovil. Je proto nutné, aby byl tok dat zastaven i programem, i když byl zastaven už předtím. Po přerušení jsou vyslány všechny bajty (může jich být až 16), které byly ve vysílacím buferu sériového portu. Takže ani při hardwarovém řízení není tok dat zastaven okamžitě.

Softwarové řízení toku dat vyžaduje kontrolu každého příchozího bajtu, aby zjistilo, zda nejde
o „zastavovací“ bajt. Dochází ke zpoždění, neboť tento bajt musí projít 16bajtovým vstupním bufe-rem. I když je řídicí bajt první, terminál čeká na naplnění buferu dalšími 15 bajty a teprve pak se začne zpracovávat obsah. Při hardwarovém řízení toku dat k takovému zpoždění nedochází.

Řízení toku ETX/ACK, resp. ENQ/ACK. Je zastaralé?

Toto je také softwarové řízení toku dat a vyžaduje speciální ovladač. Bajty jsou posílány v pake-tech (přes asynchronní sériový port), přičemž pakety jsou ukončeny řídicím znakem ETX (End of Text). Jakmile terminál dostane znak ETX, počká, až je schopen přijmout další paket, a pak pošle znak ACK (Acknowledge). Když tento znak dostane počítač, pošle další paket. A tak dále. Nevím, zda je toto řízení podporováno v Linuxu?? Stejné schéma používají některé terminály HP, avšak místo znaku ETX posílají ENQ.

Fyzické spojení

Terminál může být k počítači připojen kabelem, prostřednictvím modemu anebo přes terminálový server. Tokem dat může být buď přímá posloupnost bajtů (jako ze sériového portu) nebo pakety přenášené po síti (např. TCP/IP).

Multiportové V/V karty (adaptéry)

Můžete si pořídit sériové karty s několika sériovými porty, kterým se říká „multiportové desky“. V tomto návodu se o nich nezmiňujeme, avšak některé z nich jsou uvedeny v Návodu k sériovým portům (včetně odkazů).

Přímé připojení sériovým kabelem

Nejsnáze připojíte terminál k počítači přímým napojením na sériový port počítače. V této kapito-le jsou uvedeny i některé informace o vzájemném propojení počítačů (přes sériový port). Většina stolních počítačů má jeden nebo dva sériové porty, z nichž jeden může sloužit pro připojení myši. Pro port EIA-232 potřebujete nullmodemový kabel (kabel počítač-počítač), kterým propojíte vysí-lací a přijímací vodiče. V terminologii ethernetových sítí se tento kabel nazývá „propojovací kabel“ (propojení sériových portů ethernetovým kabelem by však nefungovalo). Chcete-li používat hard-warové řízení toku dat, pravděpodobně pro něj použijete pin DTR (anebo DTR i DSR).

Měli byste se přesvědčit, že máte správný kabel. Vyhovět by měl obyčejný nullmodemový kabel zakoupený v běžné prodejně počítačů, pokud je dostatečně dlouhý. Může být také označen jako „kabel pro sériové tiskárny“. Na skladě jej však mívají jen větší prodejny. Nesmí to být kabel k modemu s přímým propojením vodičů (které nejsou propojeny křížem). Viz kapitola „Zakoupit, nebo si udělat?“. Přesvědčte se, že se připojujete k sériovému portu DB25 nebo DB9 („samec“), nikoli k paralelnímu portu („samice“ DB25).

Číslování pinů

Čísla pinů jsou obvykle vylisována v plastu přímo u pinů. Možná si je budete muset prohlédnout v silnějším světle nebo pod lupou. Když se díváte na pin samčího konektoru DB obráceného širší řadou pinů nahoru, pin vlevo nahoře má číslo 1 (pin č. 0 neexistuje). Sousední pin má č. 2 atd. Na konci této řady je pin 5 nebo 13. Další pin (6 nebo 14) je v další řadě zcela vlevo pod pinem

1. Když se díváte na samičí konektor s delší řadou směrem nahoru, pin č. 1 je v horním pravém rohu.

Rozložení pinů na kabelu nullmodemu (3, 4 nebo 5 vodičů)

Tato tři schémata se vztahují ke skutečnému textovému terminálu. Pokud však zaměníte RTS za DTR a CTS za DSR, můžete je použít k propojení 2 počítačů. (K propojení PC nepoužívejte čtyř-vodičový kabel.) Pokud používáte na terminále řízení toku pouze DTR (jednosměrné), můžete vyřadit vodiče od RTS až po DSR. Nepoužíváte-li hardwarové řízení toku dat, můžete také vyřadit vodiče od CTS po DTR. A máte-li 2 kroucené dvoulinky, můžete 2 vodiče použít improvizovaně k uzemnění signalizace kroucené dvoulinky. Máte-li na počítači konektor DB25, musíte propojit:

PC, samec DB25 Terminál, DB25TxD Transmit Data 2 --> 3 RxD Receive DataRxD Receive Data 3 <-- 2 TxD Transmit DataSG Signal Ground 7 --- 7 SG Signal GroundCTS Clear To Send 5 <--20 DTR Data Terminal ReadyRTS Request To Send 4 --> 6 DSR Data Set Ready

Máte-li na počítači konektor DB9, zkuste propojit:

PC, DB9 Terminál DB25RxD Receive Data 2 <-- 2 TxD Transmit DataTxD Transmit Data 3 --> 3 RxD Receive DataSG Signal Ground 5 --- 7 SG Signal GroundCTS Clear To Send 8 <--20 DTR Data Terminal ReadyRTS Request To Send 7 --> 6 DSR Data Set Ready **

Máte-li na sériovém portu i na terminále konektor DB9:

PC, DB9 Terminál DB9RxD Receive Data 2 <-- 3 TxD Transmit DataTxD Transmit Data 3 --> 2 RxD Receive DataSG Signal Ground 5 --- 5 SG Signal GroundCTS Clear To Send 8 <-- 4 DTR Data Terminal ReadyRTS Request To Send 7 --> 6 DSR Data Set Ready **

Uspořádání shora nemá řídicí modemové vodiče, aby bylo možno pro getty zajistit volbu „local“ (což je ekvivalentní s stty clocal). Také když provozujete hardwarové řízení toku, musí být akti-vováno na počítači (volba -h v příkazu agetty, ekvivalent stty crtscts).

Standardní rozložení pinů na kabelu nullmodemu (7 vodičů)

Následující 3 schémata znázorňují plný „standard“ nullmodemových kabelů. Tak může být pro-pojen zakoupený kabel. Jiné rozložení je, když propojíte piny 20 a 6 a dále pin 8 s piny 4 a 5 sou-časně. Kabely s těmito rozloženími nelze použít k hardwarovému řízení toku dat (RTS/CTS) mezi přímo propojenými počítači. Obě rozložení jsou však určena jak pro softwarové řízení toku dat (Xon/Xoff) mezi terminálem a počítačem, tak i pro neřízený tok. Žádný z uvedených kabelů nelze použít k hardwarovému řízení toku dat, neboť zatímco většina terminálů podporuje řízení DTR, resp. DTR/DSR, Linux je dosud (r. 2000) nepodporuje.

PC samec DB25 Terminál DB25 DSR Data Set Ready 6 <--| DCD Carrier Detect 8 <--|- 20 DTR Data Terminal Ready TxD Transmit Data 2 ----> 3 RxD Receive Data RxD Receive Data 3 <---- 2 TxD Transmit Data RTS Request To Send 4 ----> 5 CTS Clear To Send CTS Clear To Send 5 <---- 4 RTS Request To Send SG Signal Ground 7 ----- 7 SG Signal Ground DTR Data Terminal Ready 20 -|--> 8 DCD Carrier Detect
|--> 6 DSR Data Set Ready

Alternativně, kabel pro plný přenos souborů DB9-DB25 (nullmodem, nebude fungovat s terminály, viz shora):

PC DB9 Terminál DB25 RxD Receive Data 2 <---- 2 TxD Transmit Data TxD Transmit Data 3 ----> 3 RxD Receive Data
|--> 6 DSR Data Set Ready DTR Data Terminal Ready 4 -|--> 8 DCD Carrier Detect SG Signal Ground 5 ----- 7 SG Signal Ground DCD Carrier Detect 1 <--| DSR Data Set Ready 6 <--|- 20 DTR Data Terminal Ready RTS Request To Send 7 ----> 5 CTS Clear To Send CTS Clear To Send 8 <---- 4 RTS Request To Send RI Ring Indicator 9 (není nutný)

(Ano, piny 2 a 3 mají na konektorech DB9 a DB25 skutečně opačný význam!) Vzájemné propojení dvou

DB9 (není určeno pro DTR):

PC DB9 DB9 RxD Receive Data 2 <----- 3 TxD Transmit Data TxD Transmit Data 3 -----> 2 RxD Receive Data
|--> 6 DSR Data Set Ready DTR Data Terminal Ready 4 --|--> 1 DCD Carrier Detect GND Signal Ground 5 ------ 5 GND Signal Ground DCD Carrier Detect 1 <--| DSR Data Set Ready 6 <--|-- 4 DTR Data Terminal Ready RTS Request To Send 7 -----> 8 CTS Clear To Send CTS Clear To Send 8 <----- 7 RTS Request To Send RI Ring Indicator 9 (nepoužito)

Dvě propojení shora zajistí plnou řídicí signalizaci pro modemy a pravděpodobně lze i nastavit stty -clocal. Pak je ovšem nutno zapnout terminál (kladný signál DTR) ještě před tím, než by se mohl pomocí getty atd. obvyklým způsobem otevřít port. Pokud byste jej nezapnuli, mohou nastat potíže (viz kapitola „Předčasné spuštění getty“). Z tohoto důvodu je lepší používat implicitní stty clocal (ignoruje řídicí signály modemu) a další propojení v tomto kabelu se nepoužijí.

Dříve, když byly s ignorováním modemových signálů problémy, se používal následující trik: V konektoru na straně počítače se propojily RTS a CTS a dále DSR, DCD a DTR. Když pak počítač potřebuje úvodní signál pro navázání spojení, pošle jej (falešně) sám sobě.

Co když je kabel příliš dlouhý

Je-li kabel delší než zhruba 50 stop (15–20 m), může při vysokých přenosových rychlostech dochá-zet k chybám. Někdy mohou fungovat bezchybně i mnohem delší kabely, zejména při nižších rychlostech, má-li kabel nízký kapacitní odpor nebo je-li elektronika na příjmu dostatečně citlivá. Za ideálních podmínek může fungovat přenos 9 600 baudů i na kabelu o délce 300 m. Jednou z možností, jak překlenout větší vzdálenost, je nainstalovat si poblíž sériových portů linkové zesi-lovače, provést konverzi ze symetrického signálu na asymetrický (a naopak) a pak použít krou-cenou dvoulinku. Linkové zesilovače jsou ovšem poměrně drahé.

Jinou možností, jak překonat dlouhou vzdálenost, je pokusit se odrušit co nejvíc magnetických polí, která jsou ve vysílacích a přijímacích datových vodičích vytvářena vedením elektřiny: TxD a RxD. Je nutno uzemnit proudovou smyčku, kterou protéká přibližně stejný proud (avšak opač-nými směry) a je umístěna poblíž datových vodičů. Kroucená dvoulinka odstraňuje magnetické pole nejlépe. Některé terminály DEC mají pro tento účel dva zemnicí vodiče. Jeden pár může být TxD a SG(TxD), kde SG je země. Pokud používáte páskový kabel, přesvědčte se, zda vodiče TxD a SG(TxD) jdou vedle sebe. Podobně i pro RxD.

Pokud je pro počítač i pro terminál k dispozici pouze jeden zemnicí vodič, může být pro tento účel napojen na oba vodiče kroucené dvoulinky. Můžete si to představit tak, že zpětný proud se stejnoměrně rozdělí mezi dva zemnicí vodiče, čímž se ovšem odstraní jenom asi polovina mag-netického pole. Je to ovšem lepší způsob, neboť zpětná smyčka vyhledává cestu s nejnižší impe-dancí. Návratová cesta datového signálu (např. TxD) má (díky nejnižšímu indukčnímu odporu) nejnižší impedanci, pokud se vrací stejnou kroucenou dvoulinkou. I když jsem neviděl výsledky žádných experimentálních testů této metody, měla by fungovat na větší vzdálenost.

Kabely pro hardwarové řízení toku dat

Pokud se rozhodnete využívat hardwarové řízení toku dat, pravděpodobně si budete muset vyrobit (nebo objednat výrobu) vlastní kabel. Je-li ke konektorům přístup, můžete si je předrátovat, viz kapitola „Instalace DB konektorů“. Musíte se rozhodnout, jestli budete používat pin DTR nebo jiný (jiné). Vodítkem vám může být nabídka k nastavení, neboť může obsahovat volbu pro aktivaci „DTR handshaking“ (nebo „flow control“), z níž plyne, že používá pin DTR. Anebo to může být pin DSR. Podrobnosti viz kapitola „Hardwarové řízení toku“. Starší terminály nemusí řízení toku vůbec mít.

Tipy na kabely

Obyčejný „přímý“ kabel nebude fungovat, pokud jej nepoužíváte jenom jako prodlužovací kabel nullmodemového (překříženého, resp. pro přenos souborů) kabelu nebo adaptéru. Přesvědčte se, zda lze konektory na kabelu zasunout do konektorů na hardwaru. Někdo může používat telefon-ní kabel, který má nejméně 4 vodiče (a pravděpodobně i kroucenou dvoulinku). Nejlepší je stí-něný kabel s nízkým kapacitním odporem.

Improvizace s kroucenou dvoulinkou

Viz také kapitola „Co když je kabel příliš dlouhý“. I když žádný signál EIA-232 není přizpůsoben kroucené dvoulince, stojí za pokus ji vyzkoušet. Jeden pár použijete pro vysílání, druhý pro pří-jem. Uzemnění signálu propojte vždy s jedním vodičem z každého páru. Do požadovaného vodi-če jde pouze část signálu, ale může to stačit. Z důvodu nižšího kapacitního odporu obvodu vytvo-řeného z kroucené dvoulinky (ve srovnání s uzemněním proudové smyčky) se vyšší zemnící proud omezí pouze na požadovaný pár více, než kolik vychází z teoretických výpočtů. To platí zejména u vyšších kmitočtů, neboť induktivní impedance roste s kmitočtem. Obdélníková vlna sériového portu obsahuje vyšší harmonické.

Zemnění

Pin 1 (z DB25) by měl být zemněním kostry (i celkovým uzemněním), avšak u levných sériových portů nemusí být propojen vůbec s ničím. Devítipinový konektor nemá uzemněnou kostru. Základním signálem je pin 7 a je obvykle propojen s kostrou. To znamená, že část proudu signá-lu prochází zemnícími vodiči v budově, což je nežádoucí. Stínění kabelu může být uzemněno pouze na jedné straně kabelu, avšak uzemnění na obou stranách je vhodnější, neboť je lepší, když proud protéká stíněním než zemnícími vodiči v budově??

Napojení modemu

Pomocí kombinace terminál-modem (bez počítače) můžete volat jiný počítač. Do poloviny 90. let byla v USA značně rozšířená služba BBS, kterou bylo možno volat. Na některé „vývěsky“ BBS bylo dokonce možné se napojit přes Internet. Právě v konkurenci s Internetem však tato služba neobstála.

Vytáčení z terminálu

Vedle přímého napojení terminálu (nebo počítače, který jej emuluje) na jiný počítač kabelem je možné se napojit na počítač prostřednictvím telefonní linky (anebo soukromé linky určené pro tento účel) s modemy na obou stranách této linky. Terminál (nebo počítač) obvykle vytočí číslo počítače, k němuž se chce připojit.

Většina lidí používá pro připojování PC a modem. PC může mít terminál připojený sériovým por-tem a obsluha terminálu vytáčí číslo pomocí počítače. Připojení terminálu přímo k modemu je slo-žitější, neboť terminál není příliš inteligentní a uživatel má k dispozici jen omezený počet mož-ností. Terminály mohou mít uloženy v paměti několik telefonních čísel, z nichž některé se napří-klad po stisknutí určité funkční klávesy odešle do modemu jako zpráva. Telefonní čísla mohou být uložena i přímo v modemu. Před odesláním čísla musí modem odeslat určitou iniciační posloupnost. Když na takovou zprávu obdrží odpověď od modemu z druhého konce telefonní linky, počítač, který je k němu připojen, už může spustit přihlašovací program getty.

Volání terminálu

Počítač se systémem Linux může naopak volat terminál. Volající pošle přihlašovací prompt a vola-ný účastník se může přihlásit. Na první pohled se může zdát nelogické, jak může hloupý terminál (nepřipojený k počítači) přijmout příchozí volání, avšak může.

Jedním z důvodů je, že hovorné někdy bývá účastníkům účtováno různými sazbami. Aby mohl být terminál volán, je nutno jej pří-slušně nastavit. Modem připojený k terminálu musí být nastaven na automatickou odpověď (je-li například registr S0 = 2, modem odpoví na druhé zazvonění). Když očekáváte volání, modem a terminál zapnete. Jakmile přijde hovor, vypíše se vám prompt a můžete se přihlásit.

Centrální počítač, který volá váš terminál, postupuje poněkud neobvykle. Jakmile váš modem odpoví, je nutno spustit přihlášení (getty). Centrální počítač spustí program „callback“ (někdy se jmenuje „cb“). Program callback udělá to, že z počítače A zavolá počítač B, B zavěsí a ihned zavo-lá zpět počítač A. Tak je to nutné udělat, když na počítači A emulujete terminál. V případě sku-tečného terminálu je situace složitější, neboť centrální počítač může použít pouze „zadní“ část pro-gramu callback. Soubor, který slouží k nastavení programu callback na centrálním počítači, musí být správně zkonfigurován. Callback zavolá terminál a program mgetty pak spustí na tomto portu přihlášení. Program mgetty samotný (situace na počátku roku 1998) byl původně určen pouze ke zpracování příchozích hovorů, avšak probíhají úpravy v tom smyslu, aby převzal i funkce pro-gramu callback a byl schopen provádět i odchozí hovory. Počátkem roku 1999 tyto úpravy ještě nebyly dokončeny.

Telnet

Telnet je program, který umožňuje připojení textového terminálu (nebo konzoly PC) k centrální-mu počítači po síti. Pro připojení telnetem se nepoužívají sériové porty. Pochopitelně, když sedí-te u skutečného textového terminálu, jste připojeni po sériové lince. Když však provozujete tel-net, váš počítač se k jinému počítači připojí nesériovým telnetem.

Telnet pracuje s pakety tcp/ip a může používat různé sítě: Internet, LAN atd. Telnet provozujete jako klient a připojujete se k telnetovému serveru nebo k jinému počítači na síti. Poté se vypíše přihlašovací prompt a přihlášení probíhá stejně jako po kabelu nebo sériovým portem.

Připojení k terminálovému serveru Co je terminálový server?

Terminálový server slouží k připojení několika terminálů k centrálnímu počítači (k centrálním počítačům) prostřednictvím sítě. Současné terminálové servery bývají poblíž centrálního počítače, nebo jsou přímo jeho součástí. Pokud jste ovšem k PC připojeni některými terminály napřímo nebo po sériových portech pomocí vytáčecích modemů na obou stranách, terminálový server nepotřebujete.

Když však jsou terminály připojeny k centrálnímu počítači po síti, terminálový server asi budete potřebovat kvůli konverzi sériového přenosu na síť. To je vhodné zejména pro zařízení jako tis-kárny nebo terminály, které nemají vestavěnou síťovou podporu. Definice „terminálový server“ však zahrnuje i případ, kdy jdou po síti veškerá data (s výjimkou počítače samotného), aniž by se používaly sériové porty. Pojem „terminál“ zahrnuje i tenkého klienta s uživatelským grafickým roz-hraním (GUI). Obvykle jde o sítě tcp/ip nebo ppp (dvoubodové spojení), avšak někdy jsou pod-porovány i jiné protokoly a jejich konverze.
Jedním z možných způsobů připojení „terminálu“ (resp. konzoly PC) je program telnet, který běží na vašem počítači (pochopitelně připojeném k síti). Bývaly doby, kdy byly terminálové servery jednoúčelové počítače, které se nedaly použít k ničemu jinému. Dnes jsou servery obvykle sou-částí počítačů, k nimž je možné mimo jiné připojit řadu terminálů.

Většina dnešních terminálových serverů pracuje nikoli s textovými terminály, nýbrž s tenkými kli-enty. Příkladem je „Linux Terminal Server Project“. Pochopitelně mohou být připojeny i textové terminály prostřednictvím telnetu. Nejčastěji to však bude monitor PC, který emuluje terminál typu „Linux“, a terminálový server je jen software, který běží na centrálním počítači. Telnetový server je vlastně jednoduchý terminálový server.

Počítač, který má pouze přímo připojené terminály (anebo připojené pomocí modemu bez tcp/ip nebo ppp), se také někdy nazývá „terminálový server“. I když v zásadě provádí totéž co skutečný terminálový server, přesně řečeno to terminálový server není.

Vývoj „terminálového serveru“

Původně byl terminálový server určen pro skutečné textové terminály připojené přes sériové porty. Servery skutečných textových terminálů mívaly mnoho sériových portů. Uživatel se nejdří-ve na tomto serveru přihlásil, pak se prostřednictvím tcp/ip připojil k centrálnímu počítači, kde se přihlásil podruhé. Někdy bylo první přihlášení automatické anebo uživatel dostal na vybranou, ke kterému počítači nebo ke které tiskárně se může přihlásit anebo jaký chce používat protokol.

Když byly sálové počítače nahrazeny osobními, postupně zanikalo i používání terminálů. PC však může terminál emulovat (např. pomocí minicomu (Linux) nebo (hyper)terminálů MS) a bylo možno se prostřednictvím modemů přihlašovat k BBS apod. K přihlašování sloužila celá baterie modemů, přičemž každý z nich byl připojen k sériovému portu, který byl buď na multiportové kartě nebo na jednoúčelovém terminálovém serveru. Všimněte si, že v žádném ze shora uvede-ných příkladů se nehovoří o klientském softwaru. Nejde totiž o model klient/server.

S příchodem Internetu se na telefonních linkách začala uskutečňovat dvoubodová spojení, i tak ovšem realizovaná pomocí modemů a terminálových serverů poskytovatelů připojení k Internetu. Osobní počítač však už neemulovaly textové terminály, nýbrž se zobrazovaly obrázky prohlížeče. Poskytovatelé připojení dnes od telefonních společností dostávají jen digitální signál, takže už nepotřebují skutečné modemy. Z „terminálových serverů“ se staly „servery pro vzdálený přístup“, méně často nazývané i

„digitální terminálové servery“. Poznamenejme, že připojení přes modem 56 k vyžaduje, aby měl k dispozici od telefonní společnosti digitální připojení.

Jakmile byly zavedeny servery pro vzdálený přístup, místo mnoha samostatných telefonních linek je počítač připojen už jen několika kabely, po nichž se uskutečňuje množství digitálních telefo-nických hovorů současně. U serverů pro vzdálený přístup už nenajdete změť konektorů jednotli-vých terminálů, resp. modemů, a k následníkovi terminálového serveru vlastně už nemůžete při-pojit textový terminál.

Později, s příchodem tenkých klientů, se pojem „terminálový server“ začal používat pro počítač, který slouží k připojování tenkých klientů, a to jak v systému Linux, tak i v MS Windows.

Typy konektorů a adaptérů

Konektor je ke kabelu, resp. k hardwarové jednotce, připojen víceméně napevno. V sériové komu-nikaci existují dva základní typy konektorů: 1. Typy DB s piny (DB9 a DB25) a 2. modulární tele-fonní konektory.

Adaptér vypadá přibližně jako konektor, avšak má dva konce s piny. Je to vlastně kabel s velmi krátkou kabelovou částí – zbyly pouze dva různé konektory na koncích. Adaptér se jen zasune oběma konci do jiných konektorů. Umožňuje propojení dvou nekompatibilních konektorů anebo spojení kabelových vodičů jinak než napřímo. Adaptér lze nahradit speciálním kabelem (vyrobeným obvykle vlastnoručně).

Pohlaví konektorů/adaptérů

Konektory a adaptéry na jednom konci jsou buď samci nebo samice. Konektory, které mají piny
– kolíky, jsou samci a ty, které mají zásuvky (někdy se jim ovšem také říká piny), jsou samice. U modulárních konektorů se obnažené kontakty nazývají zástrčky, vnitřní kontakty (které nejsou vidět) jsou zásuvky. Zástrčky jsou samci; zásuvky jsou samice.

Typy adaptérů

Existují tři základní typy adaptérů: nullmodem, měnič pohlaví a adaptéry portů. Některé adaptéry splňují více než jednu z těchto funkcí.

Adaptér typu nullmodem: přesměrovává vodiče jako nullmodemový kabel.
Měnič pohlaví: mění pohlaví konce kabelu tak, aby bylo možno propojit dva konektory stejného pohlaví.
Adaptér portu: přechod od jednoho typu konektoru k druhému (z DB9 na DB25 atd.).

Konektory DB

(Montáž konektorů DB na konce kabelů viz kapitola „Montáž konektorů DB“.) Mají 9 nebo 25 pinů. Specifikace EIA-232 uvádí 25 pinů, avšak vzhledem k tomu, že na sériových portech se vět-šina z nich nepoužívá, stačí 9 pinů. Rozložení viz DB9-DB25. Číslování pinů uvidíte, když se podí-váte hodně zblízka nebo přes lupu.

Modulární konektory RJ

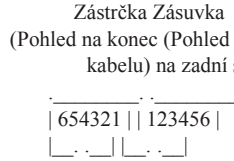
Zkratka RJ znamená „registered jack“. Vzhledově se tyto konektory nijak neliší od moderních telefonních konektorů, avšak nemusí s nimi být kompatibilní. Viz také Instalace konektorů RJ. Pro sériové porty se používají 6 nebo 8vodičové. Některé jsou i 10vodičové, avšak ty asi oficiálně nepatří do řady RJ.

6vodičové: RJ11/14, RJ12 a MMJ

RJ11 mají stejnou velikost, avšak mohou mít 2, 4 nebo 6 vodičů. Mají-li dva vodiče, nazývají se RJ11. Mají-li 4 vodiče, někdo je nazývá RJ14. Mají-li 6 vodičů, často se nazývají RJ12 (avšak tele-fonní RJ12 má jen 4 vodiče). Vypadá to zmateně, avšak mají stejnou velikost a liší se pouze počtem vodičů.

Téměř stejně vypadá konektor MMJ (6 vodičů) používaný u novějších modelů terminálů VT (a jiných). Jsou někdy označovány jako DEC-423 nebo DEC RJ11. MMJ má distanční výčnělek a není kompatibilní s RJ (ledaže by se výčnělek uříznul). Některé konektory však byly navrženy tak, aby byly kompatibilní současně s MMJ i RJ. Vzhledem k tomu, že MMJ se používají zřídka a pravděpodobně jsou i drahé, někdo si upravuje RJ (šestivodičový) tak, že uřízne výčnělek, aby byl kompatibilní s MMJ.

Rozložení pinů MMJ (DEC) vypadá takto: 1-DTR, 2-TxD, 3-TxD_Gnd, 4-RxD_Gnd, 5-RxD, 6-DSR. Cyclades Cyclom-8Ys RJ12 má: 1-DTR, 2-TxD, 3-Gnd, 4-CTS, 5-RxD, 6-DCD. Specialix IO8+ má: 1-DCD, 2-RxD, 3-DTR/RTS, 4-Gnd, 5-TxD, 6-CTS. Piny RJ (a MMJ) jsou číslovány podobně jako RJ45.



```
    |__| |__|
```

Standardní kabel MMJ pro přenos souborů (nullmodem) má konektory MMJ na obou koncích. K PC jej připojíte pomocí adaptéru MMJ-->DB. Tento adaptér zasunete do konektoru DB (řekně-me 25pinového) na zadní straně PC a konektor MMJ zasunete do něj. Pokud takový adaptér nemá-te, můžete si sami vyrobit kabel s MMJ (nebo souborovým RJ) na jednom konci a konektorem DB na druhém konci.

Standardní kabel pro přenos souborů (nullmodem) se dvěma konektory MMJ (nebo RJ11/14) pro-pojí: 1-6, 2-5 a 3-4. Poznamenejme, že tento kabel podporuje řízení toku DTR/DSR, které proza-tím není podporováno v Linuxu. Velice jednoduše si můžete zhotovit vlastní standardní 6vodičo-vý kabel pro přenos souborů, pokud víte, že obyčejný 4vodičový telefonní kabel, který vede v milionech domácností od zdi k telefonu, je také kabel pro přenos souborů. Opatřete si takový a svůj kabel můžete zapojit podle něho.

Položíte-li tento kabel rovně na podlahu (tak, aby nebyl zkroucený), zjistíte, že zástrčky na obou koncích mají zlaté kontakty směrem nahoru (anebo obě směrem dolů). Je tedy symetrický, avšak když se nad tím trochu zamyslíte, i tak může posloužit k přenosu souborů. Stačí propojit několik těchto kabelů pomocí kabelových spojek, neboť spojky jsou také nullmodemové adaptéry, tj. adaptéry pro přenos souborů. Dva kabely pro přenos souborů spojené za sebou vytvoří přímé propojení.

Uvádíme schéma kabelu (autorem je Mark Gleaves), kterým připojíte MMJ k 9pinovému sériové-mu portu a může sloužit k řízení toku dat RTS/CTS:

```
  DEC MMJ PC (Linux) DB9 Pin Signál Signál Pin ====== ====== ====== ======
1 DTR ----------------|---> DSR 6
                                                    |---> CTS 8
2 TxD --------------------> RxD 2
3 SG (TxD)-------------|--- SG 5
4 SG (RxD)-------------|
5 RxD <-------------------- TxD 3
6 SR <-----------------|--- RTS 7
                                                    |--> DTR 4
                                                     |--> CD 1
                        (bez propojení) RI 9
```

8vodičové a 10vodičové

RJ45 a RJ48 jsou 8vodičové modulární telefonní zástrčky. Údajně existují 10vodičové konektory, které jsou širší a 8vodičové do nich nejdou zasunout. Někdo je označuje jako RJ45 a/nebo RJ48, avšak toto označení nemusí být správné. Používají se jak pro ploché telefonní kabely, tak i pro kulaté kroucené dvoulinky. Zakončení může být u těchto dvou kabelů různé. RJ48 má výčnělek navíc, takže nejde zasunout do džeku RJ45 (zatímco zástrčku RJ45 do konektoru RJ48 zasunout lze). Používají se u některých multiportových sériových karet a sítí. Zde jsou čísla pinů osmivodičového kabelu:

```
Zástrčka Zásuvka
(Pohled na konec (Pohled do zásuvky
    kabelu) na zadní stěně)
    ._____. ._____.
    | 87654321 | | 12345678 |
    |__. .__| |__. .__|
       |____| |____|
```

Je lepší kabel zhotovit, nebo upravit? Koupit, nebo zhotovit?

Můžete zkusit zakoupit krátký, nullmodemový kabel. Samotný „modemový kabel“ fungovat nebu-de. Nullmodemové kabely byly svého času označovány jako kabely k „sériovým tiskárnám“, jež se už používají velice málo (2004). Mohou být také označeny jako kabely pro „přenos souborů“ nebo kabely „počítač-počítač“. Hardwarové řízení toku dat bude fungovat při spojení PC-PC (emu-lace terminálu), nikoli však řízení DTR u většiny textových terminálů. Podporuje-li Linux řízení DTR, skutečné terminály budou fungovat správně. Přesvědčte se, zda konektory na koncích kabe-lu lze zasunout do počítačových a terminálových konektorů.

Co když ale potřebujete pro napojení terminálů nebo hardwarové řízení toku delší kabely? Hotové kabely se shánějí obtížně (možná je najdete na Internetu), zejména když chcete používat co nejméně vodičů (řekněme třeba 4). Jednou z možností je, že si je necháte vyrobit, což bývá dost drahé, ledaže byste našli někoho, kdo by vám je vyrobil za cenu blízkou ceně hotových kabelů (já jsem našel).

Levná alternativa je, když si koupíte použité kabely (máte-li kde). Pokud získáte použitý terminál, vyžádejte si i kabely. Další možností je, že si vyrobíte vlastní. Možná budou malou úpravu pro-pojení potřebovat i použité kabely. V každém případě budete potřebovat speciální nástroje. Konektory krátkých kabelů jsou většinou pevně zalité v plastu a nelze je předrátovat, změnit zapo-jení můžete jen u kabelů vyrobených na zakázku nebo vlastnoručně. K výhodám vlastnoruční výroby kabelů patří, že když se to jednou naučíte, může se vám to hodit, třeba když kabel fun-guje špatně, přestane fungovat úplně anebo nahonem potřebujete

zhotovit nový kabel.

Čísla pinů 9 a 25pinového konektoru

Čísla pinů bývají vylisována v plastu konektoru, možná je však bez lupy nepřečtete. Místo DCD bývá někdy napsáno pouze CD. Číslování pinů na samičím konektoru se čte zprava doleva, při-čemž 1 je v pravém horním rohu (zatímco 1 u samčího konektoru je vpravo nahoře). Směr --> je od PC.

\1 2 3 4 5/ Pohled na piny \1 2 3 4 5 6 7 8 9 10 11 12 13/\6 7 8 9/ samčího konektoru \14 15 16 17 18 19 20 21 22 23 24 25/

Jak namontujete kabelové konektory DB

Stručný popis viz kapitola „Konektory DB“. Bohužel, většina kabelů, které dnes zakoupíte, má konektory zalité na obou koncích, a nelze je upravovat. Jen menší část má rozebiratelné konektory, které je možné předrátovat. Než se pustíte do úpravy kabelů, je třeba, abyste věděli něco

o pinech. Jsou dva typy pinů: pájené a krimpované.Krimpované piny vyžadují speciální krimpovací kleště a také „boxera“ na

zasouvání a vysouvání.
Práce s těmito nástroji je ovšem rychlejší než pájení. Napojujete-li dva vodiče na jeden pin (což je také někdy potřeba, když třeba chcete propojit vodič s jiným pinem), je pájení rychlejší (u těchto pinů). To proto, že krimpované piny mohou mít napojení pouze po jednom vodiči, zatímco k jed
nomu pinu může být připájeno vodičů několik. Krimpovaný pin pouze zatlačíte rukou nebo boxerem. Když pracujete s boxerem, musíte nejdříveuchopit vodič hrotem tohoto nástroje a pak jím obtočit zadní stranu pinu.

Odstraňování pinu boxerem vyžaduje malý trik. Lépe byste jej pochopili, kdybyste při čtení těch-to pokynů měli před sebou jak boxera, tak i vodiče. Hrot boxera vsuňte podél vodiče do otvoru co nejhlouběji (asi 1,5 cm). Na některých boxerech je značka (např. dírka), abyste věděli, do jaké hloubky jej zasouváte. Hrot boxera má zužující se výřez, který nasunete na vodič širší stranou. Některé nástroje mají dva hroty. Vodič snáze odstraníte tím hrotem, který se na něj nasouvá obtíž-něji, neboť těsněji přilne k vodiči a úchop je pevnější.

Máte-li hrot patřičně zasunutý, jemně zatáhněte za nástroj i za vodič. Když nejde ven, boxer není ve správné poloze. Zkuste jej zatlačit víc dovnitř nebo pootočit do jiné polohy (anebo obojí). Možná byste také měli použít jiný hrot, který pasuje na pin těsněji. Pomocí tohoto nástroje můžete snadno změnit přímý kabel na nullmodemový kabel (kabel pro přenos souborů) atd.

S boxerem mohou být i určité problémy. Nejde-li nasunout zezadu na pin, může to být proto, že pin nebyl správně krimpovaný k vodiči, je třeba zdeformovaný (není kulatý) apod. Pokud je povy-sunutý jen částečně, avšak nejde vytáhnout celý, pravděpodobně je ohnutý. Prohlédněte si jej pod zvětšovacím sklem. Můžete se jej pokusit narovnat pinzetou, avšak musíte postupovat opatrně, abyste nepoškodili pozlacení. Někdy se podaří pin vytlačit ploškou silnějšího šroubováku (nebo něčím podobným). Nesmíte ovšem tlačit příliš silně, abyste nepoškodili otvor v plastu nebo neo-hnuli pin.

Pájet sami raději nezkoušejte, pokud s tím nemáte žádnou zkušenost nebo jste si o tom něco nepřečetli.

Instalace konektorů RJ

Jde o určitý typ telefonních modulárních konektorů, který se používá pro běžné telefony. Existu
je ovšem mnoho různých typů (viz kapitola „Modulární konektory RJ“).Není snadné pracovat s použitými konektory. Možná se vám podaří vytáhnout vodiče, vsunoutmalý klínek, který zvedne pozlacené kontakty, a konektor bude možné znovu používat. Existujík tomu speciální krimpovací nástroje, pro každý konektor jiné.

Pokud nemáte krimpovací nástroj, můžete se pokusit o instalaci pomocí šroubováčku (a kladív-ka), která je s těmito nástroji ovšem složitější. Zatlačte na vodiče kabelu a poté silně stlačte pozla-cené kontakty šroubováčkem, který zajde do jednotlivých rýh mezi kontakty. Pokud šroubováček nebude mít plošku přibližně stejně silnou jako kontakty anebo pokud šroubováček při stlačování po kontaktu sklouzne, můžete jej poškodit. Na šroubováček také můžete poťukat kladívkem, nej-dříve jej ale zkuste zatlačit rukou.

Přesvědčte se, zda jste při stlačování kontaktů na konektoru nepoškodili západku. Konektor při vytlačování kontaktů nepokládejte přímo na stůl, nýbrž na podložku (silnou asi 1 mm), která pohodlně zajde do štěrbiny mezi západkou a tělem. Podložkou může být například silnější lepen-ka, několik vizitek nebo kousek dřeva. Vzhledem k tomu, že spodní strana konektoru (na niž konektor pokládáte) není kvůli západce zcela rovná, neměl by být povrch stolu příliš tvrdý a měl by být pokrytý také například lepenkou. Ještě lepší je podložit 6 mm dlouhou část konektoru na straně s kontakty dalším kouskem lepenky. V každém případě je vhodný měkký povrch stolu. Také je údajně možné upnout konektor do svěráku, avšak tuto metodu nemám vyzkoušenou. Pozor, aby konektor ve svěráku neprasknul.
Ve srovnání s krimpovacími kleštěmi je instalace popsaná shora mnohem delší a náchylnější k nezdarům, vyžaduje však určitou vynalézavost a je pochopitelně levnější, neboť kupovat speciální nástroje kvůli jednomu nebo dvěma konektorům je rozmařilost.

Nastavování (konfigurace) obecně

Konfigurace (nastavování) se skládá jak z uložení konfigurace do permanentní paměti terminálu, tak i z uložení příkazů do startovacích souborů (na pevném disku), které se provádějí při každém zapnutí počítače (anebo přinejmenším při každé změně). V této kapitole naleznete přehled kon-figurace základních komunikačních voleb terminálu i počítače. Další dvě hlavní části podrobně popisují konfiguraci terminálu (viz kapitola „Nastavování terminálů“) a počítače (viz kapitola „Podrobnosti o nastavování (konfiguraci) počítače“).

Přehled nastavování (konfigurace) terminálů

Při instalaci terminálu je nutno do permanentní paměti (která není po vypnutí vymazána) tohoto fyzického terminálu uložit jeho charakteristiku, která bude platná i po opětovném zapnutí terminálu. Možná vás potká to štěstí, že váš terminál bude už zkonfigurovaný, takže si ušetříte práci.

Existují dva základní způsoby konfigurace terminálu. Buď se k němu posadíte a postupně pro-cházíte konfigurační nabídku nebo na něj z centrálního počítače pošlete řídicí sekvence. Ještě však než začnete s posíláním, je nutno nastavit volby komunikačního rozhraní (viz popis), např. pře-nosovou rychlost apod. To lze provést pouze interaktivně od terminálu, který si bude povídat s počítačem, viz kapitola „Nastavení terminálu“.

Přehled nastavování (konfigurace) počítače

Ještě před zahájením posílání řídicích sekvencí z počítače na terminál je nutno také zkonfigurovat počítač samotný tak, aby byl schopen pracovat s terminálem. Budete-li mít štěstí, možná bude sta-čit jen přidat do souboru /etc/inittab příkaz getty tak, aby se při zapnutí počítače na terminál poslal prompt „login:“. Podrobnosti viz také kapitola „Getty“.

Počítač komunikuje s terminálem pomocí sériového ovladače (je součástí jádra), který má impli-citní konfiguraci a je také částečně (někdy i zcela) zkonfigurován programem getty ještě předtím, než spustí „login“ na jednotlivých terminálech. Někdy však je nutno provést pomocí programů stty a setserial dodatečnou konfiguraci. Tyto programy (pokud je to nutné) je nutno spustit po kaž-dém startu počítače, neboť tato konfigurace se po vypnutí neuchovává. Viz „Podrobnosti o nasta-vení (konfiguraci) počítače“.

Mnoho voleb

Můžete vybírat z velkého počtu konfiguračních voleb. Především musí být nastaveny správně komunikační volby, jinak počítač nebude fungovat. Pokud se některé funkce nebudou využívat, na nastavení jejich voleb nezáleží. Když například nemáte k terminálu připojenou tiskárnu, není vůbec důležité, jak jsou v terminálu nastaveny její konfigurační parametry. Poslední tvrzení ovšem není zcela korektní. Předpokládejme, že nemáte tiskárnu, avšak počítač omylem pošle terminálu příkaz, aby všechny znaky (všechna data) přesměroval z počítače pouze na tiskárnu. Na terminá-lu se nic neobjeví a bude mrtvý. Některé terminály mají konfigurační volbu, na jejímž základě informují terminál, že tiskárna není přítomna. V takovém případě bude terminál příkazy k pře-směrování na „tiskárnu“ ignorovat a problém uvedený výše nenastane. To ale situaci nijak zvlášť nepomůže, neboť existuje řada jiných chybných příkazů, které mohou být poslány na tiskárnu a které mohou způsobit škody. To se stává například tehdy, když omylem pošlete na terminál

binární soubor.V některých případech nezpůsobí nesprávné nastavení žádné problémy, dokud nespustíte nějakýmálo používaný program, který počítá s určitým nastavením terminálu. Jiné volby ovládají pouzevzhled displeje a terminál bude pracovat i tak, avšak nebude na něj pěkný pohled.

Některé volby se týkají jen terminálu a není je nutno nastavovat na počítači. Například: Chcete psát černými písmeny na světlém pozadí? Takový způsob namáhá oči méně než psaní na černém pozadí. Má se klávesa opakovat, když ji přidržíte? Má se řádek zalamovat, když přesáhne přes pravý okraj? Mají klávesy při stisknutí vydávat zvuk?

Volby komunikačního rozhraní

Některá z těchto komunikačních nastavení (voleb) jsou pro terminál i pro počítač společná a musí být ve shodě: rychlost, parita, bity/znak, řízení toku. Jiné komunikační volby jsou nastaveny pouze na terminálu (a jenom některé jsou důležitá pro ustavení komunikace). Ještě jiné volby jako adre-sa a přerušení (IRQ) fyzického portu ttyS2 jsou nastaveny jenom na počítači příkazem setseri-al. Dokud nejsou kompatibilně nastaveny všechny komunikační volby, nelze mezi terminálem a počítačem uspokojivě (anebo vůbec) komunikovat. Na terminálech je nutno tyto volby nastavit ručně pomocí nabídky (anebo pomocí určité speciální kazety). Centrální počítač je zkonfigurován po každém zapnutí do sítě (anebo když se někdo přihlásí). Někdy se o konfiguraci počítače posta-rá program getty (je uložen v souboru /etc/inittab), který zahajuje proces přihlašování do systému. Viz popis getty.

Nastavení počítače a terminálu, která musí být ve shodě:

Rychlost (bity/s).
Parita.

Počet bitů ve znaku.
Řízení toku.

Některá základní nastavení terminálu samotného jsou:

Výběr portu.
Nastavení komunikace na plný duplex (=FDX na terminálech Wyse).

Pokud program getty nenastaví počítač, jak potřebujete, můžete použít jeden z dvojice programů stty a setserial (nebo oba).

Rychlost

Na počítači i na terminále musí být nastavena stejná přenosová rychlost. Jednotkou rychlosti jsou bity/s (bity za sekundu, bps, baudy). Nastavte nejvyšší možnou rychlost, při níž ještě nedochází k chybám. Řízení toku umožňuje nastavení vyšších rychlostí. Na terminále mohou existovat dvě rychlosti: vysílací (transmit) a přijímací (receive), které jsou někdy zkráceně označovány písmeny T a R. Obvykle bývají nastaveny na stejnou hodnotu, neboť linuxový program stty dosud neumí nastavit tyto rychlosti rozdílně. (Pro nastavení rychlosti existuje v příkazu stty volba, avšak nejspíš nastavuje obě rychlosti na stejnou hodnotu.) Obvyklé rychlosti jsou 300, 600, 1 200, 2 400, 4 800, 9 600, 19 200, 38 400, ... Nižší rychlosti (např. 600) mají tiskárny a terminály, které píší na papír.

Mám používat kontrolu parity?

Definice viz kapitola „Co je to parita“. Implicitní je zablokování kontroly parity. Chcete-li kontrolu parity aktivovat, musíte vybrat, zda sudou nebo lichou. Jakou, je v podstatě jedno. U terminálu je někdy nutno nastavit paritu vysílací i přijímací. V takovém případě ji budete muset nastavit stejnou, počítač nepovolí rozdílné nastavení. Navíc sériový port PC obvykle nepodporuje nasta-vení různých parit. U některých terminálů nelze nastavit přijímací paritu a terminál bude paritní bity ignorovat. Pokud na některých starších terminálech používáte osmibitové bajty, kontrola pari-ty není možná, neboť na přenos paritního bitu už není místo.

K čemu je vůbec parita? Je vhodné ji kontrolovat, i když kontrola není povinná. Bez kontroly pari-ty můžete občas přijmout nesprávný znak a snažíte se opravovat překlepy, které ve skutečnosti neexistují. Parita nicméně něco stojí. Za prvé je složitější nastavení, neboť implicitní nastavení je obvykle bez parity. Za druhé, parita snižuje rychlost přenosu, neboť na každý bajt musíte přenést
o jeden bit víc. Ve skutečnosti přenášení paritního bitu může a nemusí snížit rychlost přenosu. Rychlost terminálu, který píše na papír, závisí na mechanické části tisku. Zvýšíte-li přenosovou rychlost, výsledkem je jenom vyslání vyššího počtu „stop“ bitů, aby terminál stačil vypsat přená-šené znaky. V důsledku delšího čekání způsobeného řízením toku není skutečná rychlost bez kontroly parity vyšší než s kontrolou. U některých terminálů je situace podobná: Zavedete-li kontrolu parity, méně se čeká v důsledku kontroly toku dat, což teoreticky vede k vyšší průměrné pře-nosové rychlosti, avšak vinou kontroly parity zůstává průměrná rychlost přenosu stejná. Existuje i možnost nainstalovat terminály bez kontroly parity a v případě potřeby ji doplnit. Chyby
v přenosu bez kontroly parity zjistíte nejlépe pomocí případných překlepů, o nichž jste přesvěd-čeni, že jste je neudělali. Když takový překlep najdete, obnovte obrazovku (znovu ji vyšlete z počí-tače), a když překlep zmizí, jde pravděpodobně o chybu parity. Dochází-li k většímu počtu tako-vých chyb (tj. více než jedna na několik set obrazovek), je nutno něco opravit, například: Akti-vujte kontrolu parity a snižte přenosovou rychlost; nebo použijte kratší a lepší kabel. Aktivace kon-troly parity nesníží počet chyb, avšak budou přesně identifikovány.

Je možný i opačný přístup, kdy nejdříve zavedete kontrolu parity a když po určitou dobu (řek-něme měsíc nebo dva) nedochází k chybám (na obrazovce se neobjevují chybové symboly), můžete kontrolu parity bez obav o bezpečnost zablokovat.

Bity/znak

Tato hodnota udává počet datových bitů na znak bez paritního bitu. Na mezinárodní znakové množiny potřebujete 8 bitů. Pokud je ovšem terminál nezná, není taková množina k ničemu. Viz kapitola „Znakové množiny“. Chcete-li používat pouze ASCII znaky, stačí 7 bitů a vysílání bude probíhat rychleji. Některé velmi staré terminály podporují pouze sedmibitové znaky.

Jaké řízení toku dat?

Vybrat si můžete mezi „hardwarovým“ (např. dtr/cts) nebo „softwarovým“ (Xon/Xoff) řízením toku dat. I když hardwarové řízení může být rychlejší (jsou-li pro tento účel v kabelu k dispozici jeden nebo dva vodiče a podporuje-li je terminál), ve většině případů Xon/Xoff funguje taktéž spolehli-vě. Někdo sice tvrdí, že kvůli rušení (viz dále) musel přejít na hardwarové řízení, avšak v mnoha případech softwarové řízení funguje bez problémů (například i v mém případě).

Používáte-li softwarové řízení toku dat (Xon/Xoff) a někteří uživatelé o tom nevědí, mohou na centrální počítač náhodně poslat znak Xoff a uzamknout si terminál. Po dobu uzamčení většinou zběsile mlátí do klávesnice a pokoušejí se terminál odemknout. Když konečně přijde Xon a komu-nikace se obnoví, vše, co v afektu napsali, se provede, často i s nečekanými následky. Při hard-

warovém řízení se podobná věc nemůže stát. Další vysvětlení viz kapitola „Řízení toku dat“.

Výběr portu

Vzhledem k tomu, že většina terminálů má na zadní straně dva i více konektorů, obvykle je jeden z nich určen pro spojení s počítačem a druhý pro připojení tiskárny. U konektoru může být označení (zjistěte to) a toto jméno (např. Aux, Serial 2 nebo Modem) může být přiřazeno buď počítači nebo tiskárně (apod.).

Rychlý pokus

I když vše, co zde bylo dosud popsáno, se může jevit jako velmi složité, uvedení terminálu do chodu bývá ve skutečnosti poměrně snadné. V části „Rychlá instalace“ naleznete jednoduchý postup, jak si můžete instalaci vyzkoušet. Pokud ovšem neuspějete nebo chcete, aby displej vypadal nebo pracoval lépe, nevyhnete se dalšímu studiu.

O konfiguraci terminálu podrobně

S výjimkou následující podkapitoly o posílání řídicích sekvencí na terminál jsou v této kapitole uvedeny zejména podrobnosti o tom, jak nastavíte terminál ručně pomocí nabídky přímo z ter-minálu. Pokud jste to již neudělali, měli byste si přečíst kapitolu „Přehled nastavování terminálů“. Nejlepší je mít k terminálu manuál; když jej však nemáte, naleznete zde informace o mnohých vol-bách, které určitě budete potřebovat.

Komunikační parametry, např. rychlost přenosu apod., musí být u terminálu nastaveny vždy, jinak s ním není možné komunikovat. Jakmile je komunikace ustavena, zbytek konfigurace můžete pro-vést dvěma různými způsoby. Buď můžete pokračovat v ruční konfiguraci u terminálu a výsledek uložit do permanentní paměti terminálu nebo můžete po každém zapnutí terminálu (resp. podob-né akci) posílat řídicí sekvence z počítače.

Nejlepší způsob je, když víte, jak správně nastavit a uložit konfiguraci v terminálu. Pokud to neví-te, obvykle stačí, když před začátkem práce na terminálu na něj ze souboru terminfo pošlete úvod-ní řetězec. Terminál lze někdy používat, i když neuděláte vůbec nic. Aplikační program i obsluha mohou nastavení kdykoli změnit prostřednictvím určité řídicí sekvence, kterou pošlou na terminál.

Posílání řídicích posloupností na terminál

Jakmile je ustavena komunikace, zbytek konfigurace terminálů může někdy být dokončen poslá-ním řídicích sekvencí z počítače na terminály. Máte-li větší počet terminálů, je dobré si napsat (nebo najít) skript, kterým konfiguraci zautomatizujete. Může (ale nemusí) obsahovat příkaz, kte-rým sdělíte terminálu, aby si uložil stávající nastavení do permanentní paměti tak, aby se mohlo použít při příštím zapnutí terminálu.

Existuje jednoduchý a složitý způsob, jak tyto řídicí sekvence rozeslat. Při jednoduchém způsobu nemusíte řídicí sekvence vyhledávat, nýbrž stačí vydat příkaz, který automaticky najde příslušnou řídicí sekvenci v databázi terminfo a pošle ji. Bohužel, databáze terminfo nemusí vždy obsahovat všechny sekvence, které potřebujete. Proto je zřejmě lepší složitější způsob s přímým posíláním řídicích sekvencí.

Složitější metoda bude vyžadovat podrobnější manuál. Manuály ke starším terminálům obsahovaly podrobný seznam řídicích sekvencí, v nových už však většinou nejsou. Možná si budete muset koupit „programátorský manuál“ (nebo nějak tak), který se k terminálům běžně nedodává. Seznam řídicích sekvencí najdete i v kapitole „Seznam řídících sekvencí“, avšak pravděpodobně není úplný.

Konfigurační příkazy můžete na terminály posílat i bez manuálů, a to pomocí programů tput a set-term. Viz kapitola „Změna nastavení terminálu“. Na terminál také můžete poslat pouze úvodní řetězec z některé položky v terminfo, pokud jej nastaví tak, jak potřebujete, viz kapitola „Úvodní řetězec“. Pokud nechcete tyto sekvence posílat z počítače pokaždé, když terminál zapnete, musí-te nastavení nějak uložit v permanentní paměti.

Nastavování starších terminálů

U starších terminálů se podívejte, jestli jsou nad horní řadou numerických kláves štítky. Pokud tam jsou, mohou na nich být napsány funkce těchto kláves v konfiguračním režimu. Některé starší ter-minály mohou mít klávesy určené pouze ke konfiguraci. A ještě starší mají mechanické přepína-če. Někdy nejsou všechny správně označeny, avšak mohou být ukryté. Pochopitelně, pokud něco nastavíte přepínačem, je to natrvalo a nastavení nemusíte ukládat do permanentní paměti.

Jak se dostanete do konfiguračního režimu

Chcete-li na terminále vybrat konfigurační volby, musíte nejdříve zadat konfigurační režim a pak vybrat z nabídky volby (tj. zkonfigurovat), které jsou uloženy v paměti terminálu a jsou zobraze-ny na obrazovce. Při této konfiguraci terminál ani nemusí

být připojen k počítači. Jak se dostane-te do konfiguračního režimu, je popsáno v manuálu k terminálu, uvedeme pouze několik užiteč-ných rad:

Má-li terminál klávesu set-up, zkuste ji stisknout. Zkuste ji také stisknout současně s klávesou Shift.

Wyse: Nejdříve zkuste stisknout Select+Shift, pak nahraďte klávesu Shift klávesou Ctrl.
VT, Dorio: Klávesou pro nastavení může být F3. Na VT420 a pozdějších modelech může být tato klávesa naprogramována na jinou činnost, takže vypněte terminál síťovým vypí-načem. Když jej znovu zapnete, stiskněte F3 ihned, jakmile se na obrazovce objeví úvod-ní zpráva.
IBM: 3151: Ctrl-ScrollLock. Ctrl-Minus_on_Keypad (nebo jako 3151).

Po nabídce se zkuste pohybovat pomocí šipek. Výběr potvrďte klávesami Return, mezerník nebo speciální klávesou (na starých terminálech je to přepínač). Z nastavovacího režimu se dostanete pomocí položky „exit“ v nabídce (na některých starších terminálech je nutno znovu stisknout nastavovací klávesu).

Komunikační volby

Aby terminál vůbec pracoval, musí být správně nastaveny hodnoty přenosové rychlosti, parity, bity/znak. Nesprávné řízení toku může způsobit poškození nebo ztrátu dat, která jsou zobrazena na obrazovce. Základní komunikační volby (terminálu i počítače) jsou popsány jinde, viz kapito-la „Komunikační rozhraní“. Následující seznam nabízí některé názvy těchto částí a taktéž některé další komunikační volby, které lze nastavit pouze na terminálu.

Rychlost (bity/s) (baudy): 9 600, 19 200, atd.
Bez parity, sudá, lichá, značka, mezera.
Bitů na znak {resp. data}: 7 nebo 8.
- Řízení toku {Hndshk}: žádné, Xon-Xoff nebo hardwarové (DTR atd).

Řízení toku příjemcem {Rcv Hndshk} chrání data přijímaná terminálem tak, že předává signály pro řízení toku centrálnímu počítači.
Řízení toku vysílačem {Xmt Hndshk} je ochrana dat vysílaných z terminálu. Terminál dostává signály pro řízení toku (a zamyká/odemyká klávesnici). Data obsahují signály Xon/Xoff.
Počet stopbitů: 1 nebo 2. Viz kapitola „Napěťová posloupnost bajtu“.
Úroveň řízení toku {Rcv Hndshk Level} {{Xoff at...}}: Řízení toku vyšle „stop“, když se přeplní bufer terminálu.
Komunikační režim {Comm}: Plný duplex {FDX}, poloviční duplex {HDX} {{Local Echo}}, lokální režim {{Online/Local}}.
Omezení rychlosti vysílání {Xmt Lim}: Omezení rychlosti vysílání na určitou rychlost ve znacích/sec (cps), i když je nastavena vyšší přenosová rychlost (v baudech).
Omezení rychlosti funkčních kláves: Jako v předchozím případě, avšak pro funkční kláve-sy (zprávy odesílané z funkčních kláves).
Výběr portu: Určení fyzického konektoru počítače (portu).

Jak uložit nastavení

Nastavení je nutno uložit do permanentní paměti terminálu tak, aby platilo i při dalším zapnutí ter-minálu. Když se vám to nepodaří, nastavení se při opětovném zapnutí terminálu ztratí. Až se bude-te opakovaně pouštět do otravné práce s nastavováním terminálu, nejdříve si zjistěte, jak je potře-ba nastavení uložit. U novějších terminálů je příkaz k uložení nastavení součástí konfigurační nabídky, v případě starších terminálů musíte hledat v manuálu. Většinou však konfiguraci uložíte příkazem Ctrl+S.

Nastavení voleb/parametry

Volby popsané v této kapitole jsou platné v konfiguračních nabídkách většiny terminálů. Volby také někdy nazýváme parametry, vlastnosti nebo funkce. Některé volby také můžeme označit jako „režimy“. Souhrnně říkáme všem volbám „konfigurační volby“. Některé volby mohou být nahra-zeny vysláním řídicích sekvencí na terminál. Různé značky a modely terminálů mají různé volby a stejné volby mohou být pojmenovány různými názvy (všechny je zde neuvádíme). Zkrácené názvy terminálů Wyse jsou uzavřeny ve složených závorkách {...}. Názvy terminálů VT jsou uza-vřeny ve dvojitých složených závorkách {{...}}.

Emulace {charakteristiky} {{režimy terminálů}}

Většina současných terminálů umí emulovat i několik jiných terminálů. Nejvíc funkcí má k dispo-zici terminál, který běží v nativním režimu (tedy neprovádí se žádná emulace) {přirozená charak-teristika}. Někdy existují i dvě různé emulace stejného modelu terminálu. Například VT220-7 emu-luje VT220 se 7 bity/bajt, zatímco VT220-8 emuluje VT220 s 8 bity/bajt (256 možných znaků).

Starší modely terminálů mívají méně vlastností než nové. Představte si, že by někdo chtěl emulovat starý terminál, avšak i s některými novými funkcemi. V určitých případech je taková emulace možná, avšak pouze do jisté míry. Tato vlastnost je někdy označována jako {Enhance} (nebo Enhanced ??).

Volby zobrazení Velikost znakové buňky {znaková buňka}

Velikost buňky, v níž je znak zobrazován. Uvádí se v pixelech (=bodech). Čím víc bodů, tím lepší je rozlišení. 10x16 znamená šířka 10, výška 16 (16 řádků, 10 sloupců). Oproti řádu matic, kdy se uvádí řádky (výška) na prvním místě, je tento zápis opačný. Znaková buňka obsahuje také mezeru mezi sousedními znaky, takže skutečná velikost buňky, jíž jsou dány hranice znaku, je menší.

Sloupce/řádky

Obvykle 80 sloupců a 24 nebo 25 řádků. To znamená, že na jednom řádku obrazovky může být až 80 znaků. Některé terminály mají volitelných 132 sloupců, avšak není-li dostatečně velká obra-zovka, písmena jsou titěrná a špatně se čtou {{Set 132 column mode}}. Chcete-li nastavit 25 řádků, přesvědčte se nejdříve, zda je tato možnost uvedena v terminfo. Také byste měli do souboru /etc/profile přidat řádek export LINES=25 a zadat příkaz stty -F /dev/ttySx rows 25. Neda-ří-li se vám rolování obrazovky, viz kapitola „Terminál neroluje“.

Kurzor

Kurzor lze nastavit, aby vypadal jako obdélník (=blok) {Blk}. Jiné volby jsou podtržení {Line} nebo blikání. Já dávám přednost obdélníkovému kurzoru, který nebliká {Steady}. Je dost velký na to, abych ho rychle našel, a nerozptyluje mě. Zvolíte-li neviditelný kurzor (taková volba na některých terminálech existuje), zmizí, avšak zůstane na místě a když budete psát, budou zde i pak naskakovat nová písmenka.

Atributy zobrazování (kouzelné cookies)

Atributy zobrazování mohou být buď kouzelné cookies nebo bajty s atributy přiřazené všem zna-kům. Kouzelné cookies mají omezený rozsah: Platí do konce řádku, nebo po konec stránky? Lepší jsou atributové bajty (jež ve skutečnosti mohou být půlbajty = nibly, čtyři bity).

Řídicí znaky zobrazení {Monitor}

Můžeme je nazývat všelijak, např. „řídicí znaky zobrazení“. Jsou-li vypnuty (obvykle jsou), interpretují se. Jsou-li zapnuty, vidíte řídicí sekvence z počítače (které na obrazovce normálně nejsou vidět). Když tedy nejsou interpretovány terminálem, s výjimkou znaků CR LF (nový řádek) je vidíte na obrazovce. Viz kapitola „Řídicí kódy“.

Dvojnásobná šířka/výška

Některé terminály mohou zobrazovat znaky ve dvojnásobné šířce a/nebo výšce. Měníte-li řádek na dvojnásobnou šířku (DW), pravá polovina (RH, right half) je z obrazovky vytlačena a vypíše se dotaz, zda má být vymazána. „Preserve“ znamená ponechat pravou polovinu řádků o dvojité šířce. Není vyloučeno, že v režimu dvojité výšky bude potřeba poslat každý řádek dvakrát (podru-hé o řádek níž).

Inverzní zobrazení {Display} (světlé/tmavé pozadí)

Normální zobrazení jsou světlá (bílá, zelená, žlutá) písmena (popředí) na tmavém (černém) pozadí. Inverzní zobrazení {Display Light} je opačné: černý text na světlém pozadí. Tento způsob je šetrnější pro oči (pokud není v místnosti tma).

Stavový řádek

Stavový řádek je řádek na horním nebo spodním okraji obrazovky, na němž jsou zobrazeny informace o běžícím aplikačním programu. Často je nějakým způsobem zvýrazněn. Je-li stavový řádek aktivní, aplikace může poslat na terminál speciální řídicí sekvenci, která znamená, že následující text má být zobrazen na stavovém řádku. Mnohé aplikace však tuto funkci nevyužívají a místo toho simulují stavový řádek přímou adresací kurzoru. Běžný uživatel obvykle nedokáže oba způsoby zobrazení stavového řádku od sebe rozlišit.

Při změně 80/132: Smazat, nebo ponechat?

Když přepínáte počet sloupců z 80 na 132 (nebo naopak), mají být data zobrazená ve starém tvaru vymazána, nebo ponechána na obrazovce? {80/132 Clr} {{Screen Width Change}}. Na nastavení této volby nezáleží, neboť když aplikační program používá 132 sloupců, nastaví si tuto volbu dle potřeby pomocí řídicí sekvence.

Stránkové volby

Aby byl terminál Wyse schopen zpracovávat více stránek zobrazované paměti (multipage) současně, musí být patřičně nastaven.

Velikost stránky

Paměť terminálu můžete rozdělit na několik stránek, popis viz kapitoly „Stránky“ a „Stránky (defi-nice)“. Stránkovou paměť můžete rozdělit na několik stránek zvolené délky. Linuxové aplikace pravděpodobně stránkování nevyužívají, takže na nastavení v podstatě nezáleží.

Zobrazení vázané na polohu kurzoru

Paměť terminálu můžete rozdělit na několik stránek, popis viz kapitoly „Stránky“ a „Stránky (defi-nice)“. Když přemístíte kurzor ve videopaměti na místo, které v daném okamžiku není zobrazeno na displeji (např. na jinou stránku nebo na nezobrazenou část běžné stránky), měla by se zobra-zit část s kurzorem? Pokud ano, nazývá se tento způsob zobrazování „vázaný“. Při pohybu kur-zoru v rámci běžné stránky se tento způsob nazývá „horizontální“ a „vertikální“ vazba, při pohy-bu mezi stránkami je to „stránková“ vazba.

Hlášení a odpově

Terminál buď identifikuje sebe a svůj stav anebo jako odpověď na určitou řídicí sekvenci pošle předběžnou odpověď.

Odpovědní hlášení (řetězec)

V průběhu nastavování může být na počítač volitelně odeslaná krátká zpráva jako reakce na zapnutí nebo na žádost z počítače (pravděpodobně na řídicí znak ENQ neboli inquire).

Automatická odpově

Je-li nastavena, při zapojení k síti se automaticky pošle na počítač odpověď, aniž by o ni musel žádat. Čeká na tuto odpověď některý z procesů getty??

Skrytá odpově

Je-li nastavena, zpětnou odpověď neuvidí nikdo než počítač. Chcete-li ji změnit, zrušte „answerback concealed“ a nahraďte ji jinou odpovědí (původní odpověď ovšem neuvidíte).

ID terminálu {ANSI ID}

Tuto odpověď terminál pošle, když je dotázán na identitu.

Volby klávesnice Pípnutí (Keyclick)

Je-li nastaveno, při stisknutí každé klávesy se ozve pípnutí (z reproduktorku v klávesnici). Někomu jde takové pípání na nervy myslím si, že je lépe je mít vypnuté.

Velká písmena {Keylock}

Když je vypnutá klávesa Caps-Lock, písmenové klávesy by měly generovat pouze velká písme-na? Je-li nastaveno {Caps} nebo „pouze horní znaky“, při stisknutí numerické klávesy a zapnutém Caps-Lock se vypíše číslo. Symbol nad číslem vypíšete, když současně stisknete klávesu Shift. To je normální režim. Když nastavíte {Shift}, při zapnutém Caps-Lock budou všechny klávesy psát horní znaky (klávesa 5 vypíše %, aniž byste museli mít stisknutou klávesu Shift, atd.).

Automatické opakování {Repeat}

Přidržíte-li klávesu stisknutou, vypisuje se opakovaně. Tato funkce je vhodná například pro popsání celého řádku stejným znakem.

Zvonek na okraji

Když je kurzor vzdálen 8 znaků od pravého okraje, zazvoní zvonek (jako na starém psacím stroji). Tato funkce se používá velice zřídka, neboť téměř všechny editory ukončují řádky automaticky (aniž by bylo nutno zadávat Return).

Přemapování kláves

Při stisknutí klávesy se normálně do počítače posílá její ASCII kód (závisí ovšem i na klávesách Shift a Control). Na některých terminálech je možné z každé klávesy poslat libovolný kód, tj. nastavením terminálu je možno klávesnici úplně přemapovat. To může být vhodné pro některé cizí jazyky nebo pro Dvořákovo rozložení klávesnice apod., které umožňuje rychlejší psaní. Dokonce existuje i software na přemapování klávesnic (i obrazovek) pro terminály, které tuto funkci nemají. V podstatě je to ovladač zařízení, který používá pseudoterminál, viz kapitola „Mapo-vání znaků: mapchan“.

Klávesa v rohu (pouze Wyse)

Terminály Wyse mají v levém dolním rohu klávesu, kterou můžete nastavit na různé funkce. Bývá označena „Funct“, „Compose Character“, „Alt“, „Hold“ nebo „Scroll Lock“. Může mít některou z následujících voleb, starší modely ovšem nemusí mít všechny:

Hold: Nerolovat. Stisknutím zastavíte tok dat (prostřednictvím řízení toku dat) na terminál. Dalším stisknutím tok dat obnovíte.
Compose: Když tuto klávesu stisknete a následuje stisknutí jiné klávesy, klávesnice vyge-neruje určený počet předdefinovaných nelatinkových znaků.
Meta: Když kromě této klávesy stisknete ještě další klávesu, v každému bajtu se bit nej-vyššího řádu nastaví na hodnotu 1.

Existují modely, kde tato klávesa tento metaefekt uza-mkne??
Funct: Když je klávesa stisknutá při psaní, všechny alfanumerické znaky přicházející z klá-vesnice budou orámovány hlavičkovým bajtem (SOH) a ukončovacím bajtem (CR).
Kpd Compose: Je-li tato klávesa stisknuta při psaní dekadického čísla na numerické klá-vesnici (následovaného znakem Enter), pošle se toto číslo v hexadecimálním tvaru??

Numerická klávesnice nebo šipky?

Numerická část klávesnice (převážně numerické klávesy uspořádané do obdélníku v pravé části klávesnice) lze nastavit tak, aby pro potřebu některých aplikací posílala speciální kódy. Totéž platí i pro šipky. V „normálním“ režimu posílají to, co je na nich napsáno (anebo v případě šipek obvyklou posloupnost), v „aplikačním“ režimu jsou to pak speciální řídicí sekvence. V některých případech existuje i numerický režim „hex“, který je téměř shodný s normálním numerickým reži-mem s výjimkou 6 nenumerických kláves, které posílají písmena A–F. Na numerické klávesnici tedy můžete napsat například „B36F“.

Co znamenají Shift-del a Shift-bs?

Shift+del někdy posílá řídicí znak CAN, Shift+bs (backspace) někdy posílá DEL, záleží na nasta-vení.

Skenovací kódy PC

Řada terminálů může emulovat klávesnici PC tak, že místo ASCII kódů pošle skenovací kódy PC (viz návod Keyboard-and-Console-HOWTO). To se nejčastěji používá ve speciálním víceuživatel-ském systému DOS OS, v běžném DOSu to fungovat nebude, viz kapitola „Nelinuxové operační systémy“. Linuxový program, který využívá sériový port, by stěží přijal skenovací kódy. Je-li toto nejnovější verze tohoto návodu, sdělte mi prosím, zda to některé programy umějí. Domnívám se, že to umí Foxpro. V terminfo musíte definovat smsc a rmcs a pravděpodobně i pctrm.

Když používáte skenovací kódy, nejlepší je používat hardwarové řízení toku dat, protože při soft-warové kontrole dochází ke konfliktům s některými kódy (??). Pokud přece jen používáte softwa-rovou kontrolu, musí být typu XPC, neboť pro zapnutí a vypnutí používá znaky 0x65 a 0x67 a stej-né hodnoty musí být nastaveny jak na terminále, tak i na PC pomocí stty.

Alternativní znaky

Některé klávesy mohou na sobě mít alternativní znaky. Jsou-li nastaveny na „psací stroj“ („typewriter“), posílané hodnoty odpovídají psacímu stroji. Jsou-li nastaveny na něco jiného, posílají se alternativní znaky.

Význam obdržených řídicích kódů Automatický nový řádek {Newline}

V tomto případě „nový řádek“ znamená, že nový řádek začíná na levém okraji běžného řádku. V Linuxu a v C může mít „nový řádek“ (NL) odlišný význam: řídicí znak line-feed (LF) se také nazývá new-line neboli NL. To proto, že v linuxovém textovém souboru znamená znak LF „nový řádek začíná zde“, takže je označován NL. Obvykle má znak LF (NL) poslaný na terminál za násle-dek posun kurzoru dolů o jeden řádek na to místo, kde byl, aniž by se vrátil na začátek tohoto nového řádku.

Je-li nastaven automatický nový řádek, „normální“ situace uvedená výše je zrušená a po přijetí znaku LF z počítače se skutečně provede fyzický nový řádek. To je přesně to, co v Linuxu potře-bujete. Kromě toho (když je nastaven automatický nový řádek) klávesa Return (nebo Enter) pošle na počítač posloupnost CR LF (v případě Wyse a VT100, ale jak u VT420??). Vzhledem k tomu, že v souborech používá Linux znak LF jako značku „nového řádku“, stačí mu poslat jen LF (nikoli tedy CR LF). Volba „nový řádek“ se tedy používá velice zřídka. Místo toho ovladač impli-citně provede požadované změny. Je to, jako kdybyste vydali příkaz stty onlcr icrnl. Avšak nemusíte, neboť toto nastavení je implicitní.

Automatický Line Feed {Rcv CR}

Jde pouze o jiný typ „automatického nového řádku“. Když přijde znak CR (carriage return), je při-dán znak LF (line feed) a výsledkem je zobrazení nového řádku. Vzhledem k tomu, že v Linuxu jsou konce řádků označovány znakem LF, tato volba se nepoužívá.

Rozpoznání znaků del (pouze Wyse??) a null (prázdný znak)

Je-li tato volba vypnutá, terminál ignoruje znak DEL. Je-li zapnutá, znakem DEL se smaže poslední znak. Prázdné znaky jsou obvykle v každém případě ignorovány. Oba znaky, DEL i NULL, jsou používány jako výplň, viz kapitola „Výplň“.

Kam směřuje zpracování textů Zalamování řádků

Nazývá se taktéž „automatické zalamování řádků“. Co se stane, když se při psaní textu narazí na konec řádku (sloupec 80 atd.) a z počítače nepřichází znak „return“ (nebo podobný)? Je-li nasta-vena volba „line wrap“, zbytek řádku se vypíše na dalším řádku atd. Jinak se ztratí a na obrazov-ce není vidět. Aplikace by měly tuto situaci zpracovávat korektně (za předpokladu, že terminfo ví, jak je „line wrap“ nastavený), a to i když „line wrap“ nastavený není. Aplikace by měla buď dlou-hé řádky zalamovat anebo poskytnout uživateli nějaký jiný způsob, jak si prohlédnout nezobra-zené části řádků (pomocí šipek apod.). Samotné systémové příkazy (ani některé jiné situace) to ovšem nemusí umět, takže v každém případě je nejlepší mít volbu „line wrap“ nastavenou.

Většina osmdesátisloupcových terminálů zalamuje 81. znak pouze tehdy, když je tisknutelný. To pro případ, že 81. znak poslaný aplikací je „return“ nebo „newline“, zalomení je provedeno správ-ně a není nutný zásah terminálu.

Rolování

Pohyb řádků nahoru a dolů nazýváme rolování {Scrl}. Existuje i panoramatické rolování, tj. rolo-vání do stran. Při běžném rolování řádky na horním nebo spodním okraji obrazovky mizí a na opačném okraji (tedy na spodním nebo horním) se objevují nové. Existují 3 typy rolování: hlad-ké, skokové a celkové. Celkové rolování vlastně ani není rolování, neboť je při něm nahrazována stará obrazovka novou (i když některé řádky ze staré obrazovky mohou zůstat i na nové). Při sko-kovém rolování řádky naráz poskočí. U hladkého {Smth} se text pohybuje nahoru nebo dolů stej-nou rychlostí. Je-li rychlost hladkého posuvu nízká, je možno každý řádek přečíst, dokud je ještě na obrazovce (v pohybu).

Hladké rolování na pomalých terminálech bývalo užitečné, neboť rolující displej bylo možné prů-běžně číst. S vyšší rychlostí přenosu je skokové rolování tak rychlé, že se při naskakování obra-zovek ztrácí méně času. Vzhledem k tomu, že čtení rolujícího textu je pomalejší než čtení pevné-ho textu, ve skutečnosti je při hladkém rolování ztráta času vyšší.

Je-li automatické rolování {Autoscrl} zablokované, nový text z počítače musí přicházet takovou rychlostí, aby se objevoval na horním okraji displeje. Pokud starý text není mazán, nový text se (nesmyslně) míchá do starého. Je-li starý text vymazán, nový text je mimo kontext, takže je lepší mít automatické rolování aktivované.

Nová stránka?

Stránkování je popsáno v kapitole „Stránky“. Je-li běžná stránka plná (je popsaný poslední řádek), má se začít rolovat, nebo se má vytvořit nová stránka (přičemž původní stránka zůstane v pamě-ti terminálu)? Je-li nastavena volba {Autopage}, vytvoří se nová stránka. Vzhledem k tomu, že strán-ky pravděpodobně nepoužíváte, tuto volbu asi vypnete.

Funkční klávesy

Funkční klávesy jsou označeny F1, F2 atd. Na starších terminálech někdy bývaly označeny PF1, PF2 atd., kde „P“ znamenalo „programovatelná“. Na některých klávesnicích jsou obojí klávesy. Klávesy je možné naprogramovat (předefinovat) tak, že při stisknutí pošlou uživatelem definova-ný řetězec bajtů. Je možné je také snadno „naprogramovat“ pomocí nastavovací nabídky {FKey}. Na některých terminálech je možno specifikovat, kam se řetězec pošle. V „normálním“ režimu má stisknutí této klávesy stejný účinek, jako kdybyste tento řetězec napsali na klávesnici. V „lokálním“ režimu se řetězec pošle na terminál (jako kdyby byl terminál v lokálním režimu). Toho lze využít například pro posílání řídicích sekvencí na terminál, když jej chcete zkonfigurovat určitým způso-bem. Ve „vzdáleném“ režimu se řetězec pošle na sériový port počítače vždy (i když je terminál v lokálním režimu).

Volby blokového režimu

Některé volby náleží pouze k blokovému režimu. Tato volba je účinná, neboť nabízí způsob, jak snížit zatížení počítače pomocí přenosu velkých celků. Tento způsob je ovšem dosti složitý a není příliš využívaný.

Forms Display

Některé oblasti obrazovky jsou určeny pro text formulářů a jsou chráněny proti zápisu volbou „prot“ {WPRT}. Pomocí voleb lze znaky v této oblasti zobrazit matně, inverzně {WPRT Rev} nebo mohou být podtržené. {WPRT Intensity} může být nastaveno na matně, normálně nebo na bílo (tedy neviditelně).

Poslat celý blok?

Má být text chráněný proti zápisu (původní text ve formuláři) poslán na počítač jako přenášený blok? {Send All}. Je text chráněný proti zápisu chráněný i proti čtení? {Send Erasable}.

Jaká oblast?

Má se poslat celá obrazovka nebo jen rolovaná oblast? {Send Area}. Má se posílání zastavit, když dojde ke kurzoru? Je-li {Xfer Term} nastaveno na „cursor“, pošlou se jen data po kurzor.

Ukončení bloku/stránky

Který symbol je nutno připojit k bloku dat, resp. ke konci stránky? K bloku {Blk End}, ke konci stránky {Send Term}.

Zámky

Existuje řada různých typů zámků. Jedním z nich je „locked keyboard“, který používáme při říze-ní toku dat, viz kapitola „Uzamykání klávesnice“. Dalším je {Feature Lock}, kterým zakážete cent-rálnímu počítači měnit nastavení terminálů prostřednictvím řídicích sekvencí. Takový zámek ovšem může způsobit nekorektní chování terminálu, neboť ignoruje řídicí sekvence poslané apli-kačním programem. Všechny nastavovací parametry nelze uzamknout. Pokud k tomu nemáte pádný důvod, terminál nezamykejte.

Uzamčením funkčních kláves zakážete počítači měnit jejich programové nastavení. Klávesy uzamykejte jen tehdy, když potřebujete zachovat jejich nastavení.

Spořič obrazovky {Scrn Saver}

Když po určitou dobu terminál nepoužíváte, spořič obrazovky vypne (nebo ztlumí) obrazovku, čímž se prodlužuje její životnost a šetří se elektrická energie. Obrazovku obvykle oživíte stisknu-tím libovolné klávesy. Vhodné je stisknout například klávesu Shift apod., neboť klávesa s určitou funkcí může být „provedena".

Tiskárna

U terminálů Wyse není k dispozici volba {Printer Attached} na zapínání/vypínání tiskárny, neboť ji není nutno zapínat. Pokud (místo na terminál) na tiskárnu přece jen pošlete řídicí sekvenci, bude se ignorovat.
Nastavení portu tiskárny je přibližně stejné (obvykle je jednodušší) jako nastavení komunikace na hlavním portu. Tiskárna má některé specifické volby. Je sériová, nebo paralelní? Je-li paralelní, je nutno tuto skutečnost při nastavování uvést a tiskárnu připojit na paralelním portu terminálu (pokud jej má). Je nutno po skončení tisku poslat na tiskárnu volbu FF (Form Feed, posun papíru na začátek nepopsané stránky)? Je-li {Print Term} nastaveno na FF, provede se posun automaticky.

Podrobně o nastavování (konfiguraci) počítače

Konfigurace počítače pro práci s terminály se provádí změnami obsahu konfiguračních souborů. Budete-li mít štěstí, budete muset změnit pouze jediný soubor, a to /etc/inittab. Změny obvyk-le provádíte editací souboru na konzole (nebo na terminále).

Program getty (v souboru /etc/inittab) Úvod do getty

Aby mohlo po zapnutí počítače přihlášení proběhnout na sériovém portu (resp. na terminálu, který je k němu připojen), do souboru /etc/inittab je nutno doplnit příkaz getty. Při zadávání z terminálu mohou nastat určité problémy (viz kapitola „Když spustíte getty z příkazového řádku, programy se zastaví"). Jméno příkazu vzniklo z „*GETs a TTY (a terminal) going"*. Na každý termi-nál je nutno vydat samostatný příkaz getty a v každém souboru /etc/inittab je alespoň jeden příkaz getty na konzole. Najděte jej a příkazy pro terminály vložte za něj. Některé soubory už obsahují příklady příkazů pro textové terminály a jsou označeny jako komentáře („zakomentová-ny"), takže je stačí „odkomentovat" (odstranit úvodní znak #) a změnit několik parametrů.

Povolené parametry závisí na konkrétním používaném getty: Pro přímo připojené terminály

jsou nejvhodnější programy getty:

agetty někdy nazývané pouze getty): Nastavení je velice snadné. Nemá konfigurační sou-bory. Viz kapitola „agetty".
getty součást getty_ps) .

Dva nejvhodnější programy getty pro modemy (nepoužívat pro přímo připojené terminály) jsou:

mgetty: nejlepší pro modemy; použití na terminálech je problematické.
uugetty: pouze pro modemy; součást balíku getty_ps.

Jednoduché getty, když nemáte skutečný textový terminál. Většina uživatelů Linuxu, kteří mají monitor, používá jeden z nich:

1. *mingetty*
2. *fbgetty*Linuxové distribuce mohou pro textové terminály obsahovat buď ps_getty nebo agetty. Některé distribuce neobsahují žádný z nich. Bohužel, často jej nazývají „getty" a musíte zjistit, který z nich to je, neboť v /etc/inittab mají odlišné parametry. Distribuce Debian používá agetty (v balíku utilit). RedHat používá ps_getty, které naleznete v *http://www.redhat.com/swr/i386/getty_ps-2.0.7j-12.i386.html.*

Když to jinak nejde, můžete to zjistit ze spustitelného souboru (obvykle v /sbin); ps_getty má v kódu /etc/gettydefs, což vyhledáte tak, že přejdete do adresáře /sbin a zadáte:

```
strings getty | grep getty
```

Je-li getty ve skutečnost agetty, výsledkem příkazu shora nebude nic. Zadáte-li však:

```
getty -h
```

měly by se vypsat volby [-hiLmw].Pokud nemáte vhodný program getty, prověřte ostatní distribuce a pomocí programu alien pro-veďte konverzi mezi balíčky RPM a Debian. Zdrojový kód si můžete stáhnout z Getty Software(*http://ibiblio.unc.edu/pub/Linux/system/serial/getty/*).Nepoužíváte-li řídicí vodiče modemu (když například používáte minimální počet, tj. 3 vodiče: vysí-lání, příjem a signální zem), musíte to pomocí „lokálního" příznaku sdělit programu getty. Tvarzávisí na konkrétním použitém getty.

Getty se po přihlášení ukončí (může ovšem spustit dceřiný proces)

Když se přihlásíte, zjistíte (pomocí top, ps -ax nebo ptree), že proces getty už neběží. Co se s ním stalo? Proč se znovu spustí, když ukončíte shell? Zde je odpověď: Když zadáte svoje uživatelské jméno, getty jej zpracuje, vyvolá přihlašovací program login a předá mu jméno. Proces getty je nahrazen procesem login. Vyžádá si heslo, ověří je a spustí proces uve-dený v souboru hesel, což je obvykle bash shell. Je-li tomu tak, bash se spustí a nahradí proces login. Poznamenejme, že jeden proces nahrazuje druhý a proces bash shell je vlastně původně spuštěný procesem getty. Souvislosti si vysvětlíme vzápětí.

Nyní je soubor /etc/inittab nastaven tak, že je-li proces getty ukončen, spustí se znovu. Je to tak uvedeno na řádku getty. Proces getty je ovšem spuštěn, i když se ukončí procesy shell, resp. login. Proč? Oba tyto procesy nahradily getty jako dceřiné procesy, a tudíž zdědily vlastnosti usta-vené jejich předchůdcem. Při detailnějším pohledu zjistíte, že procesy, které nahrazují svého před-chůdce, mají shodné ID procesu jako původní proces. Bash je tedy vlastně procesem getty, který se maskuje číslem jeho ID. Je-li ukončen, je vlastně ukončen proces getty (i když ve skutečnosti už neběží). Výsledkem je, že je znovu spuštěn proces getty.

Když se odhlásíte, všechny procesy na sériovém portu jsou ukončeny včetně bash shellu. Totéž se může stát (je-li tato funkce aktivována), když modem vyšle na sériový port signál k zavěšení linky (změní napětí na DCD). Proces getty tedy obnoví odhlášení ze systému nebo signál k zavě-šení linky. Tento proces lze spustit vynuceně i tak, že (ručně) zrušíte bash (nebo login) pomocí klávesy „k“, když se ohlásí na monitoru, nebo pomocí příkazu kill. Pravděpodobně jej budete muset zrušit pomocí signálu 9 (který systém nemůže ignorovat).

Když spustíte getty z příkazového řádku: programy se zastaví

Program getty obvykle spouštíte ze souboru /etc/inittab, nikoli z příkazového řádku. Progra-my běžící na terminále by v takovém případě mohly být neočekávaně pozastaveny nebo zrušeny. Vysvětlíme si proč (není-li tento důvod pro vás důležitý, můžete přejít k další kapitole). Když spus-títe getty řekněme na ttyS1 z příkazového řádku jiného terminálu, např. tty1, „řídicím termi-nálem“ programu getty bude tty1, i když ve skutečnosti byl spuštěn z ttyS1. Řídicí terminál programu getty je tedy nesprávný. Jestliže však bude spuštěn v souboru inittab, řídicím terminá-lem bude ttyS1, což je správné.

I když je řídicí terminál nesprávný, přihlášení na ttyS1 funguje (neboť ttyS1 byl parametrem getty). Standardní vstup a výstup jsou nastaveny na ttyS1, i když řídicím terminálem zůstane tty1. Jiné programy, které běží na ttyS1, mohou tento standardní vstup/výstup (který je připo-jen k ttyS1) zdědit a vše bude v pořádku. Avšak některé programy se mohou chybně pokusit číst ze svého řídicího terminálu (tty1), který je nesprávný. Terminál tty1 si může myslet, že tyto pro-gramy jsou spuštěny na pozadí z tty1, takže pokus o čtení z tty1 (správně by se mělo číst z ttyS1) má za následek zastavení procesu, který se pokoušel číst. (Proces na pozadí nesmí číst ze svého řídicího terminálu.) Na obrazovce se vypíše zpráva, například „[1]+ Stopped“. V tomto okamžiku jste v pasti, neboť nemůžete komunikovat s procesem, který se pokouší komunikovat s vámi přes nesprávný terminál. Můžete pochopitelně přejít k jinému terminálu, proces zrušit atd.

agetty (může se jmenovat i getty)

Příklad řádku z /etc/inittab:

```
S1:23:respawn:/sbin/getty -L 19200 ttyS1 vt102
```

S1 je z ttyS1. 23 znamená, že úroveň procesu getty je 2 nebo 3. respawn znamená, že je-li getty (anebo proces, který jej nahradí, např. bash) zrušen, getty je automaticky obnoven. /sbin/getty je vlastní příkaz. Volba -L znamená Local (ignorují se řídicí signály modemu). -h (to v příka-zu není) aktivuje řízení toku dat (totéž co stty crtscts). 19200 je rychlost přenosu. ttyS1 znamená /dev/ttyS1 (COM2 v MS-DOS). vt102 je typ terminálu, tato hodnota se nastaví do vnější proměnné TERM. Konfigurační soubor neexistuje. Po úpravě getty zadejte na příkazovém řádku init q a vypíše se přihlašovací prompt.

Automatická detekce paritních problémů v programu agetty

Program agetty se pokusí o autodetekci nastavení parity v terminále (včetně nezadané parity), k tomu viz kapitola „Osmibitové datové bajty (plus parita)“. Používáte-li stty k zadávání parity, agetty ji okamžitě zruší, neboť nejdříve potřebuje, aby paritní bit prošel jako datový bit. To proto, že při zadávání přihlašovacího jména potřebuje zpracovat poslední bit (pravděpodobně paritní), aby mohl automaticky rozpoznat paritu. Když tedy používáte paritu, aktivujte ji pouze uvnitř tex-tového terminálu a ostatní ponechte na agetty: Rozpozná paritu a nastaví ji na počítači. Pokud terminál nerozpozná paritu, kterou obdržel, přihlašovací prompt bude zkomolený, dokud na klá-vesnici nezadáte nějaký znak, z něhož getty dokáže rozpoznat paritu. Zkomolený prompt odradí nezvané hosty od pokusů přihlásit se, což je jenom v pořádku.

S automatickou detekcí parity bývají občas problémy, neboť jakmile poprvé zadáte přihlašovací jméno, agetty spustí program login, aby dokončil přihlašování. Bohužel, program login nedo-káže určit paritu. Pokud totiž paritu nedokáže určit ani program getty, nedokáže ji určit ani login. Selže-li první pokus o přihlášení, login nabídne další pokus atd. (všechny s nesprávně nastavenou paritou). Případně po větším počtu neúspěšných pokusů o přihlášení (nebo po uply-nutí časové prodlevy) se program agetty spustí znovu a znovu se odstartují všechny přihlašova-cí posloupnosti. Program getty může při opakovaném spuštění rozpoznat paritu na druhý pokus a vše bude fungovat, jak má.

Má-li text zadaný na terminále nesprávnou paritu, program login jej nepřečte správně a není možné se přihlásit. Jestliže terminál podporuje přijatou paritu, na obrazovce budou i nadále nesmyslné znaky. Pokud getty neurčí správně paritu, obvykle před

promptem vypíše soubor /etc/issue, takže na obrazovce přibudou další zkomolené znaky.

Proč není getty schopen rozpoznat paritu podle prvního zadaného znaku? Zde je příklad: Před-pokládejme, že přijde osmibitový bajt s paritním bitem 0 (bit nejvyššího řádu) a s lichým počtem jedniček. Jaká je to parita? Lichý počet jedniček by nasvědčoval liché paritě. Ale co když je to osmi-bitový znak bez parity? To zatím není možné určit, avšak alespoň jsme vyloučili sudou paritu. Roz-poznávání parity je tedy vylučovacím procesem.

Je-li další zadaný bajt podobný prvému a také pouze vyloučí sudou paritu, stále ještě není možné určit paritu. Tato situace se může opakovat donekonečna, a když změníte přihlašovací jméno, při-hlašování selže pouze v ojedinělých případech. Najde-li agetty paritní bit roven 1, bude mít za to, že jde o paritní bit, nikoli nejvyšší bit osmibitového znaku. Předpokládá tedy, že v přihlašovacím jméně nepoužíváte metaznaky (s nejvyšším bitem rovným 1), tedy že jméno se skládá pouze ze znaků ASCII. Při přihlašování se také můžete zacyklit. Předpokládejme, že jako uživatelské jméno zadáte jen jedno nebo dvě písmena a ukončíte je znakem return (enter). Pokud by tato písmena nestačila na určení parity, login se spustí ještě před jejím určením. Občas k tomuto problému dochází, když při prvním spuštění agetty terminál není zapnutý a připojený.

Když byste se tedy zacyklili, musíte několikrát za sebou stisknout klávesu enter, dokud se nevy-píše přihlašovací prompt programu getty. Jinou možností je pár minut počkat, až uplyne časová prodleva. Pak se programem getty na obrazovce vypíše přihlašovací prompt tohoto programu a můžete se zkusit znovu přihlásit.

8bitové datové bajty (plus parita)

Tuto paritu bohužel agetty určit neumí. Od konce roku 1999 nemá volbu na zablokování auto-matickou detekci parity, proto určí paritu nesprávně. Následkem toho bude přihlašování zmatené a parita bude nastavena nesprávně. Používání parity s osmibitovými datovými bajty je tedy neschůdné.

getty (součást getty_ps)

(Většina je převzatá ze starého návodu k sériovým terminálům od Grega Hankinse.) U tohoto programu je potřeba přidat položky jak do konfiguračního souboru, tak i do souboru /etc/inittab. Zde jsou některé vzorové položky, které přidáte do konfiguračního souboru /etc/gettydefs: # 38400 bps Dumb Terminal entry DT38400# B38400 CS8 CLOCAL # B38400 SANE -ISTRIP CLOCAL #@S @L login: #DT38400

19200 bps Dumb Terminal entry DT19200# B19200 CS8 CLOCAL # B19200 SANE -ISTRIP CLOCAL #@S @L login: #DT19200

9600 bps Dumb Terminal entry DT9600# B9600 CS8 CLOCAL # B9600 SANE -ISTRIP CLOCAL #@S @L login: #DT9600

Všimněte si, že DT38400, DT19200 atd. jsou jen návěští a musí se shodovat s návěštími

v /etc/inittab.Pokud byste chtěli, program getty může při přihlašování vypsat různé informace. V příkladech,které uvádím, je to jméno systému a sériová linka. Lze přidat i další informace:

@B The current (evaluated at the time the @B is seen) bps rate.@D The current date, in MM/DD/YY.@L The serial line to which getty is attached.

@S The system name.

@T The current time, in HH:MM:SS (24-hour).

@U The number of currently signed-on users. This is a count of the number of entries in the /etc/utmp file that have a non-null ut_name field.

@V The value of VERSION, as given in the defaults file. Samotný znak „@“ vypíšete tak, že zadáte „\@“ nebo „@@“. Po skončení editování souboru /etc/gettydefs můžete ověřit syntax pomocí: linux# getty -c /etc/gettydefs

Přesvědčte se, zda v systému nejsou pro sériový port, k němuž jsou terminály připojeny, jiné kon-figurační soubory programů getty a uugetty (např. /etc/default/{uu}getty.ttyS*N* nebo /etc/conf.{uu}getty.ttyS*N*), které by mohly ovlivnit činnost getty na terminále. Pokud jsou, odstraňte je.

Proveďte změny v souboru /etc/inittab tak, aby program getty běžel na sériovém portu (údaje v souboru nahraďte údaji, které odpovídají vašemu prostředí – port, rychlost, implicitní typ termi-nálu):

S1:23:respawn:/sbin/getty ttyS1 DT9600 vt100 Restart init: linux# init q

V tomto okamžiku už by se vám měl na terminále objevit přihlašovací prompt. Možná budete ještě muset přivolat pozornost terminálu stisknutím klávesy return.

mgetty

Písmeno „m“ na začátku znamená „modem“. Tento program je primárně určen pro modemy a má--li být provozován na terminále (pokud nevyužíváte hardwarové řízení toku, jež ovšem obvykle vyžaduje kabel zhotovený na zakázku), od poloviny roku 2000 musí být rekompilován. Doku-mentace přímo připojených terminálů viz kapitola „Přímé“ v manuálu mgetty.texi.

Příklad konfigurace pro terminál viz poslední řádky souboru /etc/mgetty/mgetty.config. Dokud neřeknete „toggle-dtr no“, bude si myslet, že máte modem a bude negovat pin DTR na počítači v marných pokusech o resetování neexistujícího modemu. Na rozdíl od jiných getty se mgetty nepřipojí k terminálu, dokud někdo nestiskne některou klávesu tohoto terminálu, takže do té doby budou programy top a ps vypisovat otazník (?). Logovací soubory v adresáři /var/log/mgetty/ mohou obsahovat několik varovných hlášení, která se vztahují pouze k mode-mům a která můžete ignorovat.

Příklad, jak může vypadat jednoduchý řádek v souboru /etc/inittab:

```
s1:23:respawn:/sbin/mgetty -r ttyS1
```

Příkazy stty a setserial

K nastavení sériových portů slouží příkazy stty a setserial. Některá (nebo i všechna) potřebná nastavení stty lze provést pomocí programu getty a není nutné použít příkaz setserial, resp. není nutné použít žádný z obou těchto příkazů. Tyto dva příkazy (stty a setserial) nastavují sériový port z různých hledisek. Většinu nastavení provádí stty, zatímco setserial má na starosti nastave-ní na nejnižší úrovni, např. přerušení a adresy portů. Tato nastavení „uložíte“ tak, že tyto příkazy musí být součástí určitých souborů (shellových skriptů), jež jsou provedeny při každém spuštění systému. Distribuce Linuxu převážně obsahují skripty s příkazy setserial, méně často pak s pří-kazy stty, neboť jsou potřebné jen zřídkakdy.

Setserial

Tuto kapitolu naleznete ve třech návodech: Modem, Sériové připojení a Textové terminály. Poně-kud se od sebe liší. Máte-li laptop (PCMCIA), nepoužívejte příkaz setserial, dokud si neprostu-dujete kapitolu „Laptopy: PCMCIA“.

setserial je program, který uživatel používá ke komunikaci s ovladačem sériového zařízení. Máte-li počítač vybaven jedním nebo dvěma sériovými porty, které patří ke standardnímu vyba-vení PC, tento program obvykle nebudete potřebovat. Současná jádra systémů dokonce umějí automaticky rozpoznat i více sériových portů. Výjimkou je případ, když máte starý sériový port ISA s propojkami, nebo když jádro systému (verze 2.2 a starší) nedokáže tyto přídavné sériové porty PCI ani rozpoznat, ani nastavit.

setserial umožňuje komunikaci se sériovým softwarem. Existuje ještě jeden program, který se
jmenuje stty a který komunikuje se sériovým portem. Používá se k nastavení rychlosti portu apod. Program setserial zajišťuje konfiguraci sériového portu nižší úrovně, která souvisí s IRQ (např. 5), adresami portů (např. 3f8) apod. Hlavním problémem tohoto programu je, že neumí nastavit resp. zkonfigurovat hardware sériového portu: Neumí hardwarově nastavit IRQ ani adresy portů. Navíc, když zdánlivě ohlašuje konfiguraci hardwaru, hlášení je někdy chybné, neboť hardware nezkoumá automaticky, nýbrž až na základě zvláštního pokynu. Dokonce ani nevyhledává nové typy sběrnic a některý hardware vůbec nenajde. Přestože jinak provede tento program většinu nastavení správně, problémy se zprovozněním sériového portu mohou naznačovat, že nastavení je provedeno nesprávně.

Dříve, když IRQ a adresy portů byly nastaveny propojkami přímo na sériových kartách, program setserial se používal k tomu, aby sdělil ovladači, jak jsou nastaveny. Nyní, když se k detekci nastavení sériových portů bez propojek používají metody zásuvných modulů, program setserial se už v podstatě nepoužívá, pokud nenastanou nějaké problémy anebo pokud nepoužíváte starý hardware. Navíc, je-li konfigurační soubor používaný programem setserial nesprávný, je s ním potíž. V takovém případě ani nemá smysl zjišťovat programem setserial konfiguraci portu, neboť by pouze zopakoval nesprávnou informaci uloženou v konfiguračním souboru.

Program setserial lze někdy využít k vyhledání sériového portu. Musíte však k tomu znát adre-su portu a zadat správné volby. V případě současných portů je lepší hledat sériové porty meto-dami zásuvných modulů.

Jméno setserial je tedy pro tento program poněkud nevhodné, neboť nenastavuje ani V/V adre-su, ani IRQ v hardwaru, pouze je softwarově „nastaví“ v ovladači. Ovladač mu pak naivně věří, i když je to v rozporu s údaji, které získal metodou zásuvných modulů. Špatné je, že při takovém konfliktu ani nevypíše varovné hlášení. Vzhledem k tomu, že ovladač zařízení je považován za součást jádra, ve většině dokumentace se používá slovo „jádro“ bez jakékoli zmínky o tom, že je řeč o sériovém ovladači.

Některé distribuce (a verze) jsou nastaveny tak, že program setserial je v době zavádění systému spuštěn inicializačním skriptem (který je v adresářovém stromě /etc). Avšak konfigu-rační soubor používaný tímto skriptem může být buď ve stromě /etc nebo /var. Chcete-li set-serial spouštět při zavádění systému, budete v některých případech muset provést ještě jeden úkon. Program setserial nebude fungovat bez sériové podpory vestavěné v jádru systému nebo zavedené jako modul. Při pokusu o spuštění programu setserial může být modul zave-den i automaticky.

Program setserial sice může být nastaven tak, aby zjišťoval hardwarové adresy V/V portů a pokusil se určit typ UART a IRQ, v tomto ohledu však existují určité omezení, viz kapitolu „Zjiš-ťování“. Program neumí nastavit IRQ ani hardwarovou adresu portu

sériových portů zásuvných modulů a PCI (zatímco metodou zásuvných modulů to možné je). Také neumí přímo číst data zásuvných modulů uložená v konfiguračních registrech v hardwaru. Avšak vzhledem k tomu, že ovladač umí přečíst tyto registry a program setserial vám řekne, co si ovladač myslí, může to být správně. Anebo vám může říct, co program předtím (a pravděpodobně chybně) řekl ovladači. Neexistuje způsob, jak se to bez dalších testů dozvědět.

Sériový ovladač (pro jádro Linuxu 2.4+) hledá několik „standardních" sériových portů staršího typu, zásuvných modulů na sběrnici ISA, a všechen podporovaný hardware portů na sběrnici PCI.
Najde-li porty správně, není nutno použít program setserial. Ovladač nezjišťuje zděděná IRQ, může je mít nesprávná a nemusí zjistit nastavení starých sériových portů provedené propojkami na kartě.

Kromě manuálových stránek programu setserial zkuste také získat informace z /usr/doc/set-serial..., resp. /usr/share/doc/setserial. Tam byste měli nalézt konkrétní provedení pro-gramu setserial ve vaší distribuci Linuxu. Program samotný se sice chová ve všech distribucích stej-ně, rozdílné jsou však skripty, konfigurace těchto skriptů (včetně automatické konfigurace) a jména a umístění těchto skriptů, vše závisí na konkrétní distribuci. Je-li sériový port zásuvný, informace naleznete v jiných návodech (Plug-and-Play, Serial).

Zrušení sériového modulu

Je-li sériový modul zrušen, změny provedené programem setserial ovladač zapomene. Ve skriptu (resp. v některém pracovním souboru tohoto skriptu) však mohou zůstat uchovány pří-slušné distribuce, které skript při opětovném zavedení modulu obnoví.

Zadání příkazu setserial

Zopakujme si, že program setserial neumí v hardwaru nastavit ani V/V adresy, ani IRQ. Nasta-vuje je buď software zásuvných modulů (spouštěných ovladačem) nebo propojky u starších séri-ových portů. Dokonce ani když prostřednictvím setserial zadáte V/V adresu nebo IRQ ovlada-či, nenastaví je a předpokládá, že už byly nastaveny. Zadáte-li nesprávné hodnoty, sériový port nebude fungovat správně (pokud vůbec).

Znáte-li u starších portů V/V adresu a neznáte IRQ, můžete požádat setserial, aby se IRQ pokusil
určit. Když zadáte setserial bez parametrů, vypíše se seznam všech přípustných příkazů, avšak bezjednopísmenových voleb (např. -v jako verbose, tj. upovídaný), které se jinak běžně používají.Všimněte si, že adresu v programu setserial V/V nazýváme „port".
Zadáte-li:

```
setserial -g /dev/ttyS*
```

získáte informace o konfiguraci ovladače na portech. V mnoha případech bude u portů připsané něco, co na první pohled vypadá jako chybné IRQ a chybná adresa. Pokud však také bude výpis obsahovat text: „UART: unknown", celý řádek ignorujte, neboť na uvedené adrese není sériový port.

Přidáte-li k volbě -g volbu -a, získáte více informací, i když je nejspíš stěží někdo bude potřebo-vat (anebo jim vůbec rozumět), neboť implicitní nastavení obvykle funguje správně. Hardware je obvykle nastaven stejně, jako hlásí setserial. Máte-li však problémy, je velmi pravděpodobné, že příčina spočívá v programu setserial. Ve skutečnosti můžete spustit setserial a přiřadit mu zcela fiktivní V/V adresu portu, libovolné IRQ a typ uart, jaký si přejete. Když zadáte setserial ... příště, tyto smyšlené hodnoty se vypíší. Navíc, když je vypíšete (v horní části obrazovky) příka-zem scanport (Debian), oficiálně si je zaregistruje jádro systému. Když se pokusíte tento port použít, ovladač sériového portu pochopitelně bude fungovat nesprávně (pokud vůbec). Když tedy programu setserial zadáváte parametry, je možné „úplně všechno". Tedy, téměř všechno. Při-řadíte-li portu bázovou adresu, která už je přiřazená (např. 3e8), nemusí ji přijmout. Zadáte-li však 3e9, přijme ji. Bohužel, ve skutečnosti je adresa 3e9 už přiřazená, neboť je v rozsahu začínajícím bázovou adresou 3e8. Z tohoto příběhu plyne ponaučení, že přiřazujete-li prostředky programem setserial, musíte se vždy přesvědčit o správnosti použitých údajů.

Konfigurační soubor

Přiřazení provedená programem setserial jsou po vypnutí počítače ztracená, avšak při opětovném zapnutí počítače je možné je pomocí konfiguračního souboru obnovit. V novějších verzích mohou být veškeré změny provedené programem setserial automaticky ukládány do konfiguračního sou
boru a tento program je může později znovu použít. Uložení konfiguračního souboru závisí na konkrétní distribuci. Zkuste se podívat do startovacích skriptů ve stromu /etc/ (např. /etc/init.d/ nebo /etc/rc.d/) a přečíst si skript jménem „seri-al" nebo setserial nebo tak nějak podobně. V něm se dozvíte, kde je uložen (jsou uloženy) kon-figurační soubor(y). V distribuci Debian lze tento konfigurační soubor použít čtyřmi možnými způ-soby:

Tento soubor vůbec nepoužívejte. Sériový ovladač si při každém spuštění systému nalez-ne porty sám a program setserial se vůbec nespustí (volba „jádro").
Uložte do konfiguračního souboru to, co program setserial ohlásí při prvním vypnutí systému. Poté už v konfiguračním souboru neprovádějte žádné změny, a to ani tehdy, když někdo na příkazovém řádku zadá příkaz setserial a pak systém vypne (volba „jednorá-zové uložení").
Při každém vypnutí systému uložte do konfiguračního souboru vše, co zjistí program set-serial (volba „automatické ukládání").

Konfiguraci nastavte v konfiguračním souboru ručně pomocí editoru. Neprovádějte žádné

automatické ukládání konfiguračního souboru („manuální“ volba). V dávných dobách (asi před rokem 2000), když ještě neexistovaly konfigurační soubory, se kon-figurace prováděla ručně (tzv. pevné kódování) přímo ve skriptu, který spouštěl program setserial. Podrobnosti viz kapitola „Edice skriptu (před verzí 2.15)“.

Zjišování

Pomocí programu setserial zjišťujte port pouze v případě, když tušíte, že byl aktivován (pomo-cí metod zásuvných modulů, BIOSem, propojkami apod.). Jinak jej setserial nikdy nenajde, neboť jeho adresa neexistuje. Problém je, když software hledá port na určité V/V adrese. Ještě před zjišťováním pomocí setserial lze spustit příkaz scanport (Debian), čímž zjistíte všechny možné porty naráz. Získáte hrubý odhad toho, co je na portech, avšak nikoli IRQ, což by ovšem pro začá-tek mělo stačit. Počítač sice může zhavarovat, avšak v mém případě to funguje dobře. Jiné distri-buce než Debian pravděpodobně program scanport nepodporují. Existuje nějaký jiný skenovací program?

Program setserial s příslušnou volbou může (na dané V/V adrese) zjistit sériový port, avšak musíte znát správnou V/V adresu. Když například chcete zjistit port na /dev/ttyS2, program bude zkoumat tu adresu, o níž si myslí, že je na ní ttyS2 (2F8). Sdělíte-li programu setserial, že je ttyS2 na jiné adrese, bude zkoumat tuto adresu, atd.

Účelem tohoto zjišťování je určit, zda je na příslušné adrese uart, a když ano, jaké má IRQ. Pro-gram setserial používejte až jako poslední možnost, neboť existují rychlejší způsoby testování, např. wvdialconf k detekci modemů, sledování zpráv při zavádění systému nebo příkazy pnpdump --dumpregs a lspci -vv. Chcete-li však přece jen použít program setserial, zadejte například:

```
setserial /dev/ttyS2 -v autoconfig
```

Když jsou výsledkem zprávy, které říkají, že typ uart je např. 16550A, vše je v pořádku. Když však vypisují typ „unknown“, na této V/V adrese pravděpodobně není sériový port. Nicméně, i tak může být na této adrese sériový port, protože některé levné sériové porty se neidentifikují správně.

Kromě automatického zjišťování typu uart umí program setserial automaticky zjišťovat IRQ, avšak nikoli vždy korektně. Stává se, že nejdřív poskytne nesprávné irq, avšak když příkaz zopakujete, už najde správné irq. Ve verzích >= 2.15 může být výsledek posledního zjišťování automaticky uložen do konfiguračního souboru dané distribuce, např. v případě Debian jsou to /etc/serial.conf nebo /etc/sysconfig/serial nebo /var/lib/setserial/autoserial.conf. Při dalším spuště-ní systému se použijí.

Může se stát, že dva sériové porty budou mít v hardwaru nastavenou stejnou V/V adresu. To pochopitelně na sběrnici ISA není povolené, avšak stává se to. Zjistí se pouze jeden sériový port, i když ve skutečnosti existují dva. Mají-li však různá IRQ, výsledkem zjišťování může být IRQ=0. V mém případě, když jsem chtěl pomocí setserial přiřadit IRQ fiktivní hodnotu, to tak dopadlo.

Konfigurace v průběhu spouštění systému

Zatímco program setserial může být spuštěn inicializačním skriptem, před zaváděním sériové-ho modulu (anebo když jádro spustí vestavěný sériový ovladač zakompilovaný do jádra) běží něco, co se podobá programu setserial. Sledujete-li na obrazovce zprávy vypisované při spouš-tění systému, vypadá to, jako by program běžel dvakrát. Ve skutečnosti je tomu tak.

Týká-li se první zpráva staršího sériového portu, vypsaná IRQ mohou být nesprávná, neboť pro-gram nezjišťuje IRQ. Vypíše-li se druhá zpráva o sériových portech, údaje mohou být převzaty například ze skriptu /etc/init.d/setserial. Obvykle se žádné zjišťování neprovádí a údaje
o nastavení hardwaru jsou nesprávné. Pouze jsou vypsána konfigurační data uložená v konfigu-račních souborech. Původní metoda (před verzí setserial 2.15) spočívala v ručním zápisu dat přímo do skriptu.

Zavádí-li jádro sériový modul (anebo je ekvivalent tohoto modulu vestavěný v jádru), detekují se všechny podporované porty zásuvných modulů. U starších portů (k nimž nejsou připojeny zásuv-né moduly) se provádí automatická detekce pouze ttyS{0-3} a ovladač je nastaven tak, aby pou-žíval pouze IRQ 4 a 3 (bez ohledu na nastavení IRQ v hardwaru). Neprovádí se žádné zjišťování IRQ, avšak je možné je provést ručně. Nastavení naleznete v hlášeních vypisovaných při zavádě-ní systému ve stejné podobě, jak je vypisuje program setserial.

Chyby v IRQ mohou být opraveny programem setserial spuštěným ze skriptu (může být spuš-těn i z jiných důvodů). Bohužel, jsou-li v něm IRQ uvedena nesprávně, jádro systému je převez-me. Tento soubor je obvykle součástí inicializace prováděné při spouštění systému. Zda se pro-vádějí či nikoli, závisí na konkrétním nastavení a na úrovni běhu.

Než začnete upravovat konfigurační soubor, nejdříve byste měli příkaz setserial ve změněné podobě vyzkoušet na příkazovém řádku. Někdy se změny provedené tímto příkazem při vypíná-ní systému automaticky uloží do souboru, např. do /etc/serial.conf (nebo autoserial.conf nebo serial). Takže pokud jste odstranili chybu a systém je v pořádku, není nutno provádět v konfiguračním souboru žádné změny. Viz kapitola „Konfigurační metody“.

Edice skriptu (ve verzích starších než 2.15)

Popíšeme si situaci před verzí programu setserial 2.15 (1999). Cílem bylo ve stromě /etc modi-fikovat (případně vytvořit) skript,

který spustí program setserial při zavádění systému. Většina distribucí takový skript obsahuje (nemusí však být ve stromě /etc).

Před verzí 2.15 (1999) byla tedy situace jednodušší, stačilo změnit skript. K programu setserial nee-xistovaly žádné konfigurační soubory setserial apod. Bylo nutno najít soubor, který spouštěl set-serial při zavádění systému a změnit jej. Pokud neexistoval, museli jste si jej vytvořit (anebo při-dat příkazy do souboru, který se spouštěl při startu systému). Pokud už takový soubor existoval, býval umístěn někde ve stromě /etc. Avšak Redhat <6.0 jej měl uložený v adresáři /usr/doc/setserial/ a teprve před použitím jej bylo nutno přemístit do adresáře /etc.

Dříve se běžně používal skript /etc/rc.d/rc.serial. Distribuce Debian používala soubor /etc/rc.boot/0setserial. Také se používal soubor /etc/rc.d/rc.local, avšak ten se nespouštěl dostatečně brzy. Vypisovalo se hlášení, že se nějaký proces pokouší otevřít sériový port ještě před tím, než byl spuštěn rc.local, což vedlo k chybě komunikace. Později byl tento soubor umístěn v /etc/init.d/, avšak nebyl určen k tomu, aby se v něm prováděly změny.

Jestliže takový soubor existoval, pravděpodobně obsahoval řadu „zakomentovaných“ příkladů. Pokud jste některé z nich „odkomentovali“ (odstranili příznaky komentářů) a upravili je, mohlo se vám podařit nastavit je správně. Bylo důležité, abyste v programu setserial použili správnou cestu a platná jména zařízení. Soubor jste si mohli vyzkoušet „ručně“ (stačilo zadat jeho jméno jako superuživatel) a zjistit, zda funguje, jak má. Takové otestování bylo mnohem rychlejší než opako-vané zavádění systému.

Od verzí >= 2.15 (za předpokladu, že distributor implementoval změny, Redhat to nejprve neu-dělal) může být provádění těchto změn složitější, neboť soubor, který spouští program setserial při zavádění systému, tj. /etc/init.d/setserial nebo podobně, nebyl určen k tomu, aby se v něm prováděly změny. Viz kapitola „Konfigurační metody pomocí /etc/serial.conf“ atd.

Příklad řádku v takovém skriptu:

```
/sbin/setserial /dev/ttyS3 irq 5 uart 16550A skip_test
```

nebo, pokud chcete, aby setserial automaticky určoval uart a IRQ pro ttyS3, příkaz by mohl vypadat třeba takhle:

```
/sbin/setserial /dev/ttyS3 auto_irq skip_test autoconfig
```

Toto se provedlo pro všechny sériové porty, které měly být konfigurovány automaticky pomocí jména, které na počítači skutečně existuje. V některých případech to nefungovalo, příčinou byl hardware.

Konfigurační metody pomocí /etc/serial.conf atd.

Před verzí setserial 2.15 (1999) se konfigurace tohoto programu prováděla ručně, a to tak, že se upravil skript, který spouštěl program setserial při zavádění systému. Viz kapitola „Edice skriptu (před verzí 2.15)“. Metoda byla jednoduchá a jasná, avšak byla změněna na něco neskutečně slo-žitého. Skript a konfigurační soubor jsou dva různé soubory. Skript zůstává nezměněn, přičemž příslušná data získává z konfiguračního souboru, např. /etc/serial.conf (nebo /var/lib/set-serial/autoserial.conf).

Navíc, někdy ani není nutné měnit soubor serial.conf (nebo podobný), protože příkaz setserial zadaný z příkazového řádku může provést příslušnou změnu souboru serial.conf automaticky. Takto to bylo uděláno proto, aby nebylo nutné editovat soubory kvůli tomu, co program setseri-al nastavuje (nebo mění) před každým zaváděním systému.

Často se stává, že když vypínáte systém, skript, který spouští program setserial při zavádění systé-mu, je spuštěn znovu, avšak tentokrát provede pouze „ukončovací“ část své činnosti, která říká: Pomocí programu setserial zjisti okamžitý stav programu setserial a tento údaj ulož do konfigu-račního souboru, např. serial.conf. Když tedy spustíte program setserial, aby změnil soubor serial.conf, nezmění jej hned, nýbrž až (a pokud) normálně vypnete systém.

Nyní už asi tušíte, co se může stát. Předpokládejme, že nevypnete systém normálně (vypadne proud apod.) a změny se neuloží. Anebo předpokládejme, že experimentujete s programem set-serial a na závěr jím zapomenete obnovit původní stav (nebo obnovíte původní stav špatně). Pak se vaše „experimentální“ nastavení uloží. A nejhorší je, že nevíte-li, které volby jsou nastaveny v konfiguračním souboru, nevíte ani, co se může stát. V distribuci Debian (a možná i v jiných distribucích) existuje jedna volba nazvaná „AUTOSAVE-ONCE“, která uloží pouze změny prove-dené příkazem setserial poprvé.
Je-li nastavena volba „###AUTOSAVE###“ a ručně změníte soubor serial.conf, tyto změny se neu-chovají, neboť při vypnutí systému je soubor uveden do původního stavu. Změny souboru seri-al.conf při vypínání systému lze zablokovat tak, že z prvního řádku souboru serial.conf odstraní-te příkaz „###AUTOSAVE###“ (nebo podobný). Dříve byl v distribuci Debian příkaz „###AUTOSAVE###“ z prvního řádku po prvním vypnutí po instalaci odstraněn automaticky. Aby byl účinek tohoto příkazu zachován, byla vytvořena volba „AUTOSAVE-ONCE“, která pouze pro-vede uložení, když je systém vypínán poprvé (po instalaci nebo aktualizaci programu setserial).

Soubor, z něhož se obvykle spouští program setserial při zavádění systému (v souladu s konfi-guračním souborem), je nyní /etc/init.d/setserial (Debian) nebo /etc/init.d/serial (Redhat) atd., který by však neměl být měněn. Chcete-li, aby byl ve verzi 2.15 v distribuci Redhat
6.0 setserial spouštěn při zavádění systému, musíte přemístit soubor /usr/doc/setserial-2.15/rc.serial do adresáře /etc/init.d/. Port zablokujete tak, že jej pomocí setserial nastavíte na „uart none“, což nebude uloženo. Soubor /etc/serial.conf bude pravděpodobně mít pro každý port jeden příkazový řádek s parametry umístěnými za příkazem setserial. Nepoužíváte-li automatické ukládání, můžete editovat soubor /etc/serial.conf ručně.

Aby se stávající nastavení provedené programem setserial uložilo do konfiguračního souboru (serial.conf) bez vypínání systému, proveďte tytéž úkony, které provádíte při normálním vypnu-tí systému: Spusťte skript /etc/init.d/{set}serial stop. Příkaz „stop“ uloží stávající konfigu-raci, avšak sériové porty budou normálně fungovat dál.

Někdy lze při vypínání systému obě konfigurační metody, tedy starou a novou, kombinovat, avšak doufám, že při spouštění systému funguje jen jedna. Debian si značí staré soubory pomocí „...pre-2.15“.

IRQ

Implicitně budou mít ttyS0 a ttyS2 číslo IRQ 4 a ttyS1 a ttyS3 budou mít IRQ 3. Zatímco sdí-lení sériových přerušení (používaná v běžících programech) je na sběrnici PCI v pořádku, na sběr-nici ISA není povoleno, pokud: 1. nemáte jádro 2.2 nebo lepší, 2. neprovedli jste kompilaci s pod-porou tohoto jádra a 3. tuto funkci nepodporuje sériový hardware. Viz Návod k sériovým rozhra-ním, Sdílení přerušení a Jádra 2.2+.

Máte-li pouze dva sériové porty ttyS0 a ttyS1, je všechno v pořádku, neboť u neexistujících zaří-zení nemůže dojít ke konfliktu přerušení.Přidáte-li starý interní modem (bez zásuvných modulů) a ponecháte ttyS0 a ttyS1, měli byste sepokusit nalézt nepoužité IRQ a nastavit je na sériovém portu (nebo na modemové kartě) a pak jejpomocí programu setserial přiřadit k ovladači zařízení. Není-li IRQ 5 použito pro zvukovou kartu,lze je použít pro modem.

Laptopy: PCMCIA

Používáte-li laptop, informace o sériové konfiguraci získáte v Návodu k PCMCIA. V případě séri-ových portů na základní desce se program setserial používá stejně jako na běžném PC. Avšak u karet PCMCIA (např. u modemů) se věci mají jinak. Program setserial by se měl v konfiguraci systému PCMCIA spouštět automaticky, takže jej nemusíte spouštět. V opačném případě (např. při spuštění skriptem nebo souborem /etc/serial.conf) by mohly nastat problémy. Nastavení karet PCMCIA by nemělo být automaticky ukládáno do serial.conf (což však Debian do verze 2.15-7 dělal). Konfiguraci karet PCMCIA pomocí setserial pochopitelně zjišťovat můžete.

Stty

Program stty provede podstatnou část konfigurace sériových portů, avšak vzhledem k tomu, že to provádějí i aplikační programy (a také program getty), není nutné používat stty příliš často. Hodí se, když nastanou nějaké problémy nebo když potřebujete zjistit nastavení. Momentální nastavení zkuste zjistit tak, že na terminále, resp. na konzole, zadáte stty -a. Také když zadáte tento příkaz bez volby -a (all), získáte krátký výpis, v němž je uvedeno, čím se nastavení liší od standardního. Také se nesnažte naučit všechna přípustná nastavení, pokud se nechcete stát histo-rikem sériového připojení, neboť řada nastavení se týká pomalých předpotopních terminálů ze 70. let minulého století. Většina implicitních nastavení bude fungovat.

Program stty je podrobněji popsán v manuálových stránkách a v informačních stránkách. Zadej-te man stty nebo info stty. Zatímco program setserial pracuje pouze se skutečnými sériovými porty, program stty se použí-vá jak pro sériové porty, tak i pro virtuální terminály, např. pro standardní linuxové textové roz-hraní na monitoru. Řada nastavení stty zde nemá žádný význam. Např. změna přenosové rych-losti se neprojeví vůbec nijak.

Program stty konfiguruje například tyto položky: rychlost (bity/s), parita, bity/bajt, počet stop bitů, odstranění 8. bitu?, řídicí signály modemu, řízení toku dat, přerušovací signály, značky konce řádku, změna malá/velká, výplň, zvukový signál při přeplnění buferu?, echo psaného textu na obrazovce, povolení úlohám na pozadí, aby mohly psát na terminál?, definice speciálních (řídi-cích) znaků (např. definice přerušovací klávesy). Další podrobnosti naleznete v manuálových a informačních stránkách. Také se můžete podívat do manuálových stránek termios, které pokrý-vají stejnou množinu voleb, avšak (zhruba od poloviny roku 1999) také informace, které v manu-álových stránkách stty nenaleznete. Používání některých speciálních znaků viz kapitola „Speciál-ní (řídicí) znaky“.

Některé implementace programu getty (např. getty_ps) mají v konfiguračním souboru /etc/get-tydefs příkazy, které byste očekávali spíše v konfiguračních souborech stty. Někdy může stačit k nastavení pouze příkaz getty na příkazovém řádku i bez konfiguračního souboru, takže nepo-třebujete stty.

Je možné psát programy v C, které mění konfiguraci stty apod. Seznámíte-li se s dokumentací některých těchto programů, zajisté to přispěje k lepšímu pochopení příkazu stty (a příslušných parametrů). Návod k sériovému programování může být užitečný, avšak je poněkud zastaralý. Manuálové stránky: termios obsahuje popis struktury (typu termios) v jazyce C, která ukládá konfiguraci stty do paměti počítače. Mnohá jména příznaků ve struktuře C jsou téměř shodná (a mají stejný význam) jako parametry příkazu stty.

Volby pro řízení toku

K nastavení hardwarového řízení toku dat používejte „crtscts“. Pro softwarové řízení toku existují

3 nastavení: ixon, ixoff a ixany. Nastavení *ixany*: zejména pro terminály. Pozastavené řízení toku obnovíte stisknutím libovolnéklávesy. Zastavíte-li rolování klávesou Stop Scroll (nebo podobnou), obnovíte je stisknutím libo-volné klávesy. Obvykle ovšem nejčastěji používáte klávesu Scroll Lock.

Nastavení *ixon*: Aktivuje naslouchání portu Xoff, a když přijde, přenos se zastaví. Podobně se zna-
kem Xon přenos obnoví.Nastavení *ixoff*: Aktivuje port, aby na přenosovou linku poslal signál Xoff, když hrozí přeplněníbuferu v hlavní paměti. Chrání zařízení, na němž je umístěn port, před přeplněním. V případě
pomalého „hloupého“ terminálu (nebo jiného pomalého zařízení) připojeného k rychlému PC je zahlcení portu počítače nepravděpodobné, takže k aktivaci ixoff dochází zřídka. Obvykle je aktivován jen „pro strýčka příhodu“.

Používání stty na „cizím“ terminále

Jak použijete stty, abyste zjistili nastavení jiných terminálů než těch, které používáte? Když je tako-vý „cizí“ terminál v činnosti a běží na něm shell, obvykle není možné zjistit nastavení. V jiných případech, když například chcete zjistit nastavení ttyS2 a píšete přitom na jiném terminále (např. tty1), zadejte stty -F /dev/ttyS2 ... (nebo --file místo -F). Je-li na místě tří teček (...) volba -a, vypíší se všechna nastavení (-a jako „all“).

Když však na cizím terminále (v našem případě ttyS2) běží shell, vypsaný údaj bude pravděpo-dobně nesprávný a nastavení nebude fungovat. Tento problém se týká jak virtuálních terminálů, tak i tty3 vers. tty1 atd. Lépe jej pochopíte, když si přečtete kapitolu „Dvě rozhraní na jednom terminále“.

Dvě rozhraní na jednom terminále

Používáte-li shell (např. bash) s možností edice příkazových řádků, existují dvě různá rozhraní ter-minálu (tj. co uvidíte, když zadáte -a). Současné shelly mají dočasné „syrové“ rozhraní (resp. syro-vý režim), kdy se každý znak přečte editorem příkazového řádku tak, jak jej zadáte. Jakmile ovšem stisknete klávesu return, editor příkazového řádku se ukončí a rozhraní terminálu se změní na „hotové“ (jako „hotové“ jídlo oproti jídlu syrovému) rozhraní (resp. hotový režim). Hotový režim trvá do odeslání dalšího promptu na terminál (což je malý zlomek sekundy). Je nutno si uvědo-mit, že příkaz nikdy není zadáván v hotovém režimu, nýbrž to, co zadáte v syrovém režimu, shell čte, i když je terminál v hotovém režimu.

Jakmile je odeslán prompt, terminál přejde z „hotového“ do „syrového“ režimu (v němž je, když například spustíte editor vim). Prompt signalizuje spuštění řádkového editoru. Nastavení „syrové-ho“ režimu je odvozeno pouze ze základního nastavení „hotového“ režimu stty. Syrový režim toto nastavení převezme, avšak změní některá jiná nastavení, aby změnil režim na „syrový“, který v žádném případě není založen na předchozím nastavení „syrového“ režimu. Použije-li tedy někdo stty ke změně nastavení syrového režimu, takové nastavení bude nenávratně ztraceno po stisk-nutí klávesy return na terminále, o němž lze předpokládat, že byl „nastaven“.

Nyní, když zadáte stty, abyste se podívali na rozhraní terminálu, může se vám objevit jak syrový, tak i hotový režim. Je nutno si uvědomit, který z nich se zobrazil. Zadáte-li stty z cizího terminálu (tedy z jiného, než z toho, na kterém píšete), uvidíte syrové nastavení. Veškeré změny se budou týkat pouze syrového režimu a po stisknutí klávesy return na cizím terminále, který jste se poku-sili nastavit, budou ztraceny. Pokud ovšem zadáte příkaz stty, abyste zjistili/změnili konfiguraci ter-minálu, který právě používáte, a stisknete return, je to jiná báchorka. Terminál po stisknutí kláve-sy return přejde do hotového režimu, změny jsou uloženy a zůstanou v platnosti i po návratu do syrového režimu (pochopitelně pokud to není z hlediska syrového režimu nepřípustné nastavení).

Taková situace s sebou může přinášet problémy. Představte si, že zhavaruje rozhraní terminálu. Chcete je obnovit z jiného terminálu, zadáte tedy příkaz stty -F dev/ttyS1 sane (nebo podobný). Ale nebude to fungovat! Můžete pochopitelně zkusit zadat stty sane ... přímo na tom terminále, který zhavaroval, avšak neuvidíte, co zadáváte. Vše, co bylo řečeno výše, platí nejen pro fyzické terminály, nýbrž i pro virtuální terminály na monitoru a pro terminálová okna v systému X Win-dow. Jinými slovy, platí to skoro pro každého, kdo používá Linux.
Naštěstí všechny soubory, které spouštějí program stty v průběhu zavádění systému, pravděpo-dobně nějak souvisejí s terminály (nebo se sériovými porty bez terminálů), na nichž neběží shell, takže s tímto speciálním případem pravděpodobně nebude problém.

Kdy je nutno zadat příkaz stty?

Je-li nutné programem stty nastavovat sériové rozhraní při každém spuštění počítače, příkaz stty musíte zadat v souboru, který je vždy prováděn při startu (při zavádění systému). Měl by být spuš-těn před tím, než se začne používat příslušný port (což platí i pro program getty). Pokud běží na různých místech, měli byste si je zapamatovat, neboť může dojít ke konfliktu. Konfiguraci si zdo-kumentujte.

Program bude pravděpodobně v souboru, který při zavádění systému spouští i program setseri-al. Místo závisí na konkrétní distribuci i verzi. Zdá se, že nejlepší je umístit jej za program setse-rial, neboť úseky s nízkou úrovní jsou prováděny nejdříve. Máte-li adresáře ve stromě /etc, kde se všechny soubory provádějí při zavádění systému (System V Init), můžete si pro tento účel vytvo-řit soubor jménem „stty“.

Původní metody přesměrování

Zhruba před rokem 2000, když jste chtěli použít stty na cizí terminál, bylo nutno zadat operátor pro přesměrování <. Když jste například seděli u tty1 a chtěli aplikovat stty na ttyS2, zadali jste: stty ... < /dev/ttyS2. Po roce 2000 (verze setserial >= 1.17 a verze

stty >= 2.0) byla vytvo-řena lepší metoda s využitím volby -F: stty -F /dev/ttyS2. Když selže původní metoda přesměro-vání, tato metoda bude fungovat.

V příkladu shora na přesměrování je programu stty přiřazen jako standardní vstup ttyS2. Tím program stty získá odkaz na „soubor“ ttyS2 a může z něj „číst“. Avšak místo očekávaného čtení bajtů posílaných na ttyS2 vyhledá pomocí tohoto odkazu nastavení konfigurace portu, které může číst nebo měnit. Pokusy o nastavení terminálu typu stty ... > /dev/ttyS2 nefungovaly. Místo toho se zprávy normálně vypisované příkazem stty na terminálu, u něhož sedíte (např. tty1), posílaly na ttyS2. Avšak nastavení ttyS2 se nezměnilo.

Problém spočívá v původně používaném operátoru přesměrování (a s novou volbou -F k němu nedojde). Někdy zůstane příkaz stty viset a neděje se vůbec nic (nevypíše se prompt ani po stisk-nutí return). Je tomu tak pravděpodobně kvůli portu, který čeká na signál na jedné z řídicích linek modemu. Například pokud nenastavíte „clocal“ tak, aby ignoroval řídicí linky modemu, a jestliže nepřijde žádný signál od CD, port se neotevře a stty na něm nebude fungovat (pokud ovšem nepoužijete novou volbu -F). Zdá se, že podobná situace existuje i v případě řízení toku dat. Nemá-li kabel vedoucí k portu vodič pro pin, po němž je nutno poslat signál, odstranění nedefinovaného („visícího“) stavu je dost nesnadné.

Jedním z možných způsobů, jak takový stav odstranit, je použít novou volbu -F a příslušně nastavit „clocal“ a „crtscts“. Pokud nemáte volbu -F k dispozici, můžete zkusit spustit na portu nějaký pro-gram (např. minicom), který jej přinutí k činnosti, i když to řídicí linky budou odmítat, a můžete dou-fat, že tento program nastaví port tak, že už napříště nebude k otevření clocal, resp. crtscts, vyža-dovat řídicí signál. K tomuto účelu budete pravděpodobně muset program „minicom“ znovu zkon-figurovat, ukončit jej a znovu spustit. Pravděpodobně jednodušší by bylo restarovat celý počítač.

Původní metoda přesměrování (která funguje i v nových verzích) spočívá v tom, že zadáte příkaz stty ... < /dev/ttyS2. Jestliže nová metoda s volbou -F funguje a stará nikoli, znamená to, že port „visí“ kvůli chybějícímu signálu na řídicí lince modemu. Původní metoda tedy může být vhodná k odstraňování takovýchto problémů.

Terminfo & termcap (stručně)

Podrobnější informace o termcap naleznete v kapitole „Terminfo a termcap“. Databázi terminfo (dříve termcap) používá mnoho aplikačních programů. Databáze obsahuje pro každý model nebo typ (např. vt100) terminálu jednu položku (nebo soubor), která říká, co terminál umí, jaké kódy se na něj mají posílat pro provádění jednotlivých akcí a jakými kódy se mají inicializovat.

Vzhledem k tomu, že řada terminálů (a PC) umí emulovat jiné terminály a mají různé operační „režimy“, k danému fyzickému terminálu může existovat i více položek a je třeba si mezi nimi vybrat. Obvykle mají podobná jména. Poslední parametr getty (a to jak agetty, tak i getty_ps) obsahuje jméno terminálu (nebo terminálové emulace), který používáte (např. vt100).

Terminfo slouží i k jiným účelům než k pouhé specifikaci toho, co terminál umí, a k poskytová-ní kódů, kterými se tyto funkce vyvolají. Také může specifikovat, jak bude vypadat „tučné“ písmo (bude-li zvýrazněno inverzí nebo vysvícením apod.), jak bude vypadat kurzor, budou-li písmena černá, bílá nebo budou mít jinou barvu apod. V terminologii PC jsou tyto údaje nazývány „prefe-rence“. Také specializuje inicializační kódy, které se mají posílat na terminál (analogie k řetězcům posílaným na modemy). Tyto řetězce se z Linuxu na terminály neposílají automaticky, viz kapito-la „Inicializační řetězce“. Pokud vám způsob zobrazování a chování terminálu nevyhovuje, může-te je v databázi terminfo (resp. termcap) změnit (a později aktualizovat). Podrobnosti viz kapitola „Kompilátor terminfo (tic)“.

Nastavení TERM a TERMINFO

Pro terminály jsou určeny dvě vnější proměnné: TERM a TERMINFO, avšak je možné, že je vůbec nebudete používat. TERM musí obsahovat typ používaného terminálu (např. vt100). Neznáte-li typ (jméno), viz kapitola „Jak se jmenuje můj terminál?“. TERMINFO obsahuje cestu k databázi ter-minfo, avšak je-li databáze umístěna implicitně, tuto cestu nemusíte znát (anebo může být pro-měnná TERMINFO nastavena automaticky souborem, který je dodávaný s danou distribucí Linu-xu). Můžete také nahlédnout do kapitoly „Umístění přeložené databáze“.

Program getty naštěstí většinou nastavuje proměnnou TERM ještě před přihlášením. Použije typ terminálu zadaný na příkazovém řádku getty (v /etc/inittab). To umožňuje aplikačním pro-gramům nalézt v této databázi jméno terminálu a poté vyhledat jeho funkce. Viz také kapitola „Proměnná TERM“.

Jestliže se databáze terminfo nenajde, na terminále se vypíše chybové hlášení. Pokud se něco tako-vého přihodí, je nutno zjistit, kde je tato databáze uložena, a je-li to nutné, nastavit podle toho TERMINFO. Databázi terminfo můžete najít například tak, že se pomocí příkazu locate pokusíte najít třeba soubor vt100. Přesvědčte se, zda databáze obsahuje také váš terminál. Nastavení proměnné TERMINFO může vypadat například takto: export TERMINFO=/usr/share/terminfo (tento příkaz by měl být v souboru /etc/profile nebo v podobném). Pokud vám údaje v této databázi přísluš-né k vašemu terminálu nevyhovují, můžete je změnit, viz kapitola „Terminfo & termcap (stručně)“.

Jak se v terminfo jmenuje můj terminál?

Abyste mohli nastavit vnější proměnnou TERM, musíte znát přesné jméno terminálu, které zadáte programu getty. V databázích termcap i terminfo se používají stejná jména, takže je budete hle-dat pouze jednou. Terminály často mívají aliasy (alternativní

jména), v takovém případě použijte první z těchto jmen.
Jméno hledejte v souboru /etc/termcap... (pokud existuje). Když takový soubor v systému není, podívejte se buď do stromů terminfo (viz kapitola „Umístění přeložené databáze“) nebo zkuste najít zdrojový kód tohoto souboru (viz kapitola „Umístění zdrojového kódu databáze“).

Zřídka používaný soubor /etc/ttytype

Konfigurační soubor /etc/ttytype se v databázi terminfo používá k mapování /dev/ttySn na jména terminálů. Používá je tset, avšak je-li vnější proměnná TERM správně nastavená, nebudete tento soubor potřebovat. V jiných systémech typu Unix (např. FreeBSD) je mapování v souboru /etc/ttys mnohem rozsáhlejší, např. na příslušný příkaz getty, kategorii připojení (např. „vytá-čené spojení“) apod. Příklad řádku v linuxovém souboru ttytype: vt220 ttyS1.

Omezení při přihlašování

Implicitně se uživatel root nemůže přihlásit z terminálu. Kdybyste takové přihlašování chtěli povo-lit, musíte vytvořit (nebo upravit) soubor /etc/securetty podle manuálových stránek „securetty“. Omezení přihlašování určitých uživatelů a určitých terminálů lze docílit změnami v souboru /etc/login.access (původně to byl soubor /etc/usertty??). Soubor /etc/login.def určuje, zda má být soubor /etc/securetty vyřazen z používání. V souboru /etc/porttime jsou nasta-veny časy, kdy mohou určití uživatelé a určité terminály používat počítač. Učiní-li některý uživa-tel příliš mnoho chybných pokusů o přihlášení, může mu být další přihlašování odepřeno. Další podrobnosti naleznete v mauálových stránkách faillog.

Podmíněné zadávání příkazů (když TERM=my_term_type)

Při spouštění systému je někdy potřeba provádět určité příkazy pouze pro určitý typ terminálu. V případě příkazu stty je to jednoduché, neboť terminál v příkazu můžeme specifikovat pomocí operátoru přesměrování (<). Avšak jaká je situace v případě aliasů a funkcí? Můžete například změ-nit příkaz ls tak, aby vypisoval adresáře barevně pouze na barevných terminálech a konzolách, zatímco na monochromatických terminálech by měl příkaz se stejným jménem (avšak s jiným tělem) psát tak, aby barevné kódování nahrazoval určitými symboly. Kam je nutno definici tako-vé funkce umístit?

Můžete ji umístit do příkazu if v souboru /etc/profile, který je spuštěn při každém přihlášení. Podmíněný příkaz if definuje určité funkce apod. pouze tehdy, jde-li o terminál určitého typu.

Příklad funkce ls

I když většinu činností prováděných tímto příkazem lze provést v konfiguračním souboru dirco-lors, zde je příklad, jak to lze provést v shellu bash:

```
if [ “$TERM” = linux ]; then
    eval `dircolors`; elif [ “$TERM” = vt220 ]; then
    ls () { command ls -F $* ; }# export funkce ls():
    declare -xf ls else echo “From /etc/profile: Neznámý typ terminálu $TERM” fi
```

Mapování znaků: mapchan

Existuje volně šiřitelný program mapchan, kterým můžete podle uživatelské tabulky namapovat znaky zadané na klávesnici terminálu (vstup) na jiné znaky. Také je možné mapovat znaky posí-lané na výstup (na obrazovku). Tato funkce je vhodná například pro přemapování klávesnice na abecedu jiných jazyků. Většina distributorů ovšem tento program nepodporuje (pokud ano, dejte mi, prosím, vědět). Zdrojový kód napsal Jura Kaliničenko (Ukrajinec, avšak program je napsaný v ruštině s anglickým manuálem) a býval na adrese *http://www.kron.vinnica.ua/free/download/*, která ovšem už v roce 2003 byla mrtvá.

Terminfo a termcap (podrobně)

Úvod do terminfo

Terminfo (dříve termcap) je databáze nejen funkcí terminálů. Poskytuje aplikačním programům informace o funkcích všech (nebo téměř všech) terminálů. K dispozici jsou řídicí sekvence (posloupnosti řídicích znaků), které je nutno poslat na terminál, aby byly provedeny určité čin-nosti jako pohyb kurzoru, vymazání části obrazovky, rolování obrazovky, změna režimu, změna vzhledu (barvy, jas, blikání, podtrhování, inverzní zobrazení apod.). Zhruba od roku 1980 vznik-ly stovky různých terminálových funkcí (některé parametry nabývají i číselných hodnot).

Jedním z možných způsobů, jak mohou aplikační programy tyto informace z databáze získat, je funkce „ncurses“, použitelná v programech napsaných v jazyce C. Když například chce program posunout kurzor 6 sloupec ve třetím řádku, jednoduše zadá move(3,6). Funkce move() (součást ncurses) ví, jak toto provést na terminále (informace si přečte v terminfo). Funkce tedy pošle na terminál řídicí sekvenci příslušnou pro daný terminál. Některé programy získávají informace přímo ze souborů databáze

terminfo bez použití ncurses. Linuxový balík může tedy sám potřebovat data-bázi terminfo, i když nepoužívá program ncurses.

Zkratky v terminfo jsou většinou delší než odpovídající zkratky v termcap, takže je snazší odhad-nout, co znamenají. Detailnější informace naleznete v manuálových stránkách terminfo (jsou tam i zkratky termcap). Pokud tedy nemusíte používat termcap, zvolte raději terminfo.

Databáze terminfo

Databáze terminfo vzniká překladem, takže existuje jak jeho zdrojová část, tak i přeložená část. Původní databáze termcap má pouze zdrojovou část, avšak tento zdroj lze jediným příkazem sou-časně převést na terminfo a přeložit. Můžete se tedy klidně obejít bez zdrojového kódu terminfo, neboť databázi terminfo vytvoříte ze zdrojového kódu termcap. Jakou momentálně používáte data-bázi (a monitor), můžete zjistit pomocí příkazu infocmp, po jehož zadání se vypíše jméno zdro-jového souboru terminfo.

Zda je terminál (např. vt100) v databázi terminfo, zjistíte pomocí příkazu locate vt100. Potřebu-jete-li najít jméno terminfo pro daný terminál, projděte si výpis souborů v přeložené databázi nebo si přečtěte kapitolu „Jak se jmenuje můj terminál v databázi terminfo?“.

Kde je databáze umístěna? Umístění přeložené databáze

Zadáte-li locate vt100, může se vypsat /usr/lib/terminfo/v/vt100, /usr/share/terminfo/v/vt100, /home/.../.terminfo/v/vt100 nebo /etc/terminfo/v/vt100. V každém z těchto adresářů mohou být soubory přeložené databáze terminfo. I když adresář /etc/terminfo není standardním umístě-ním, máte-li několik typů terminálů, může být toto vhodné místo pro případ, že není k dispozici adresář /usr. Adresář /usr může například být na zvláštním disku nebo v jiné oblasti, kterou se nepodařilo připojit. Programy, které používají databázi terminfo, si ji obvykle dokáží najít samy, pokud je na jednom z uvedených míst. Jinak je cesta k ní uložena ve vnější proměnné TERMINFO, např. TERMINFO=/usr/share/terminfo.
V linuxové distribuci Debian je v balíčku ncurses-term několik běžně užívaných terminálů (včet-ně monitoru-konzoly). Jsou v adresáři /etc/terminfo/. Všechny ostatní terminály jsou v balíčku ncurses-bin a jsou uloženy v adresáři /usr/share/terminfo/.

Je-li přeložená databáze terminfo na několika místech, vše bude obvykle fungovat až do té doby, než někdo nainstaluje nové soubory terminfo (z novější distribuce, z Internetu, změnou původních souborů apod.). Všechny staré soubory totiž musí být nahrazeny novými (na všech místech), aby byly zlikvidovány veškeré možné redundance. Nejste-li si jisti, zda jste úspěšně nahradili všechny soubory, určitě se najde aplikační program, který použije starý soubor (nejspíš s chybnými daty), který ještě zůstal někde na disku. Tomuto problému lze předejít aktualizací vnější proměnné TERMINFO, jak jsme se zmínili výše.

Umístění zdrojové databáze

I když zdrojový kód nemusí být instalován na počítači, existuje způsob, jak jej můžete získat z pře
ložené databáze. Pouze zadáte příkaz infocmp. Soubor se zdrojovým kódem (pro všechny terminály) se může jmenovat /etc/termcap nebo ter-minfo.src, případně i jinak. Viz manuálové stránky: vytvoření (nebo úprava) požadovaného tvaru zdrojových souborů je popsáno v terminfo(5) ale také resp. termcap(5). Soubor terminfo.src může být umístěn na různých místech, anebo vůbec nemusí být součástí distribuce. Najít se ho můžete pokusit pomocí příkazu locate. Na Internetu je na adrese *http://catb.org/terminfo/*.

Překladač terminfo (tic)

Data ze zdrojového souboru jsou překládána programem tic (terminfo compiler), který provádí konverzi mezi formáty termcap a terminfo. Databázi terminfo tedy vytvoříte ze zdrojového sou-boru termcap. Instalační program, kterým byl vytvořen Linux, pravděpodobně nainstaloval přelo-žené soubory na pevný disk, takže nemusíte překládat vůbec nic, ledaže byste ve zdrojových sou-borech /etc/termcap (resp. terminfo.src) provedli nějaké změny. Program tic automaticky nainstaluje výsledné přeložené soubory do adresáře terminfo, kde je používají aplikační progra-my. Kde je tento program nainstalovaný, závisí na ... najděte si to raději v manuálových stránkách man tic.

Podívejte se na terminfo

Je vhodné se podívat na položku v terminfo, která odpovídá používanému terminálu (pochopitel-ně do zdrojového kódu), a pročíst si komentáře. Zběžně, bez komentářů, si můžete prohlédnout zdrojový kód pomocí příkazu infocomp. Z komentářů se však můžete dozvědět o terminálu důle-žité informace, například jak je nutno jej nastavit, aby správně pracoval s databází terminfo.

Vynechání nepotřebných dat

Když v databázi terminfo zrušíte všechna data, která se nevztahují k používaným (resp. uvažova-ným) terminálům, ušetříte místo na disku. Nemažte však data v databázi termcap označená jako „Linux terminal“ (konzola), a provozujete-li X Window, pak ani xterm. Dále se může hodit „hlou-pý“ terminál („dumb“) pro případ, když aplikační program nedokáže určit typ terminálu. Nainsta-lujete-li pouze terminfo vztahující se k připojeným terminálům (anebo k terminálům, jež hodláte připojit v blízké budoucnosti), ušetříte prostor na disku a údaje o nově připojené terminály nain-stalujete z Internetu do databáze termcap v několika sekundách.

Chyby ve stávajících souborech terminfo (a v hardwaru)

V souborech terminfo a termcap je bohužel celá řada chyb. Navíc, mnohé z těchto souborů jsou neúplné a u některých terminálů chybí určité funkce. Někdy je soubor i tak použitelný, jindy se však neobejdete bez opravy anebo si musíte najít jinou, lepší emulaci.

Neutěšený stav souborů v databázi terminfo má řadu příčin. Jednou z nich je skutečnost, že v 80. letech, kdy byla většina z nich napsána (pochopitelně ve formátu termcap), pochopitelně nevyu-žívaly tehdejší aplikační programy všech současných technických vymožeností terminálů. Na tuto skutečnost si ovšem nemůžeme stěžovat. Současné programy využívají nové funkce, např. vim pracuje s „kontextovým zvýrazňováním“, nebo minicom používá množinu grafických znaků. To ovšem vyžaduje doplnění nových definic do původního termcapu, což mělo (anebo nemohlo?) být už dávno hotové.

Terminály mají i hardwarové chyby (ve firmwaru), a když byl kvůli nim upraven termcap, byly tak vlastně „zakonzervovány“. Výrobci sice nabízeli náhradní mikroprocesory s opravenými chybami, avšak ne všichni uživatelé se obtěžovali jejich nákupem. Takže terminfo pro určité terminály se v závislosti na různých firmwarech zdvojnásobil. Tato skutečnost není v databázi termcap pod-chycena a součástí Linuxu je pouze jedna databáze. Na některé hardwarové chyby se v minulos-ti ani nepřišlo, neboť určité funkce nikdo nepoužíval. Některé uváděné chyby tedy nikdy nebyly opraveny, protože třeba nebyly důležité, výroba terminálů zanikla apod.

Úprava souborů terminfo

K této činnosti potřebujete manuál terminálu s popsanými řídicími sekvencemi. Novější manuály, tak od r. 1990, je obsahují. Potřebujete také manuál terminfo (například manuálové stránky ter-minfo). Opravený zdrojový soubor přeložte programem tic, který by měl výsledek překladu auto-maticky uložit do správných adresářů.

Chcete-li najít k určitému terminálu lepší záznam v terminfo, můžete zkusit pohledat po Interne-tu (avšak to, co najdete, nemusí být vždy lepší). Najdete-li nicméně lepší záznam, který se zdá být stabilní (tedy po určitou dobu jej bez problémů používáte), měli byste jej poslat tomu, kdo tuto databázi udržuje (bývá uveden ve zdrojových souborech).

Řetězec init

Součástí databáze terminfo bývají také tzv. inicializační řetězce, které slouží k inicializaci terminá-lu. Mohou měnit vzhled obrazovky, měnit režimy nebo spouštět emulaci na jiné terminály. Na ter-minály se žádné inicializační řetězce neposílají automaticky. Někdo by se mohl domnívat, že pro-gram getty by měl poslat inicializační řetězec. Pokud by jej poslal, mohl by přepsat stávající nasta-vení terminálu. Aplikační programy inicializační řetězce neposílají.

Inicializační řetězec musíte zadat pomocí příkazu na příkazovém řádku (resp. ve skriptu, např. v /etc/profile). Takovými příkazy jsou tset, tput init a setterm -initialize. Někdy je posílání inicializačních řetězců zbytečné, neboť terminál se umí po zapnutí nastavit sám (pomocí voleb resp. „preferencí“ uložených v permanentní paměti terminálu).

Proměnná TERM

Ve vnější proměnné TERM by mělo být nastaveno jméno používaného terminálu. Není-li nastave-no a vy je neznáte, viz kapitola „Jak se jmenuje můj terminál?“. Do této proměnné je obvykle nasta-ven obsah parametru terminal_type, který předáváte programu getty (viz soubor /etc/inittab file). Jméno musí být obsaženo v databázi terminfo. Nastavení proměnné TERM zjistíte příkazem set zadaným na příkazovém řádku (anebo tset -q). Na konzole (na monitoru) je proměnná nasta-vena na „linux“, což je monitor osobního počítače, na němž je emulován fiktivní model terminá-lu jménem „linux“. Vzhledem k tomu, že „linux“ je velmi podobný terminálu vt100 (což platí pro mnoho dalších textových terminálů), označení „linux“ může sloužit jako vhodné dočasné označe-ní textového terminálu.

Je-li k jednomu portu (/dev/tty...) připojeno několik typů terminálů (například terminál s pře-pínačem různých typů nebo port připojený k modemu, na nějž uživatelé volají z různých typů ter-minálů), proměnná TERM musí být nastavena při každém připojení k sériovému portu. Často exi-stuje řídicí sekvence, jejímž prostřednictvím se počítač může zeptat na typ terminálu. Jinou mož-ností je zeptat se přímo uživatele. K tomu lze využít buď tset anebo si napsat krátký skript.

Program tset je popsaný v manuálových stránkách. Tento program je určen ke zjišťování jména používaného terminálu. Jakmile jej zjistí, vyhledá v databázi terminfo příslušné údaje a pošle na terminál inicializační řetězec. Může také poslat hodnotu TERM. Když například uživatel vytočí číslo a přihlásí se, provede se přihlašovací skript .profile, který obsahuje příkaz eval `tset -s ? vt100`, což má za následek dotaz na uživatele, zda používá terminál vt100. Uživatel buď odpoví „yes“, nebo zadá typ terminálu, který používá. Pak teprve pošle program tset inicializační řetězec a nasta-ví do proměnné TERM jméno (typ) terminálu.

Dokumentace k terminfo/termcap

Manuálové stránky k terminfo(5) (nejlepší) a termcap(5). Druhé vydání Termcap Manual (autor Richard M. Stallman), *http://www.delorie.com/gnu/docs/termcap/termcap_ toc.html*, je manuál GNU. Je poněkud zastaralý, neboť neobsahuje terminfo.

Soubory terminfo.src a /etc/termcap obsahují informace o různých verzích souborů termcap, konvence jmen terminálů a speciální funkce kódů u6-u9. Nemáte-li jej, nalezne-te jej na *http://catb.org/terminfo/*.
Knihu Termcap a terminfo vydalo nakladatelství O'Reilly v roce 1988.

Jak používat terminál

V této kapitole naleznete informace o řízení rozhraní terminál – počítač a o změnách nastavení terminálu v průběhu jeho používání. Je zde vysvětleno (anebo jsou uvedeny odkazy na vysvětle-ní), jak má uživatel řídit a sledovat rozhraní a jak má zadávat různé příkazy ovladači terminálu (zařízení). Kapitola se nezabývá aplikačními programy, shellem ani utilitami. Dva často používa-né příkazy na terminále:

clear (vymazat obrazovku)
reset (resetování terminálu)
setterm -reset (alternativa k „resetování“ v případě chyby)

Zapnutí terminálu

Nejdříve musí být terminál pochopitelně zapnutý. Pokud nevidíte přihlašovací prompt, stiskněte několikrát return (resp. enter). Pak zadejte přihlašovací jméno (ukončené znakem return/enter), a jakmile se vypíše další prompt, zadejte také heslo (rovněž ukončené klávesou enter/return). Přesvědčte se, zda nepíšete velkými písmeny. Pokud ano, počítač se domnívá, že máte starý terminál, který neumí psát malá písmena, a ovladač se nastaví tak, aby posílal na ter-minál pouze velká písmena.
Pokud se nic neděje, přesvědčte se, zda jsou počítač i terminál v pořádku. Je-li počítač vypnutý, i tak se mohou zadané znaky objevovat na obrazovce, neboť na počítači mohou být propojené přijímací a vysílací piny a znaky se opisují i při vypnutém počítači. Nedaří-li se přihlášení ani při zapnutém počítači, viz Odstraňování závad.

Ovladač (sériového) terminálu

Když zadáváte příkazový řádek, shell (např. bash shell) čte zadávané znaky a reaguje na ně. Zada-ný znak nejdříve projde ovladačem, který je součástí operačního systému. Ovladač může určité znaky měnit (např. znak return zadaný na klávesnici může změnit na „nový řádek“ používaný v linuxových souborech). Také rozpozná určité řídicí znaky, jež můžete zadat na klávesnici, např. ^C, kterým se ukončuje program. Znaky se obvykle opisují na obrazovce. Chování terminálu lze nastavit programem stty, například lze zrušit některé (nebo všechny) funkce.

Problémy s editory

Při používání editorů vi a emacs mohou nastat určité problémy. Některé terminály nemají kláve-su escape (ESC), takže je nutno tento znak zadávat jako Ctrl+[.

emacs

Pokud se provádí softwarové řízení toku dat, příkaz ^S v editoru emacs „zmrazí“ (znehybní) dis-plej a činnost obnovíte znakem ^Q. Někdy je emacs zkonfigurován tak, že je tento znak nama-pován na jinou klávesu. Některé terminály mají metaklávesy, takže lze nastavit terminál tak, aby vytvářel metaklávesy.

vi a kurzorové klávesy

Editor vi používá klávesu esc jako příkaz k opuštění vkládacího režimu. Většina terminálů bohu-žel posílá šipky na počítač jako řídicí sekvence (začínající znakem ESC) a vi musí dokázat rozlišit tyto dva významy znaku ESC. Lepší varianty editoru vi (např. vim, je-li příslušně zkonfigurován) jsou schopny rozpoznat rozdíl podle časové prodlevy mezi ESC a další klávesou. Je-li čas krátký, znamená to, že byla stisknutá šipka. Další podrobnosti viz nápověda ke kurzorovým klávesám.

Anebo jiné řešení. Na terminálech VT může být klávesa s levou šipkou nastavena buď tak, že posí-lá sekvenci ESC [D nebo ESC O D. Ostatní šipky jsou na tom podobně, pouze místo D používa-jí A, B a C. Máte-li se šipkami problémy, nastavte si posloupnost ESC [D, neboť znak „O“ je v jiné souvislosti chápán jako příkaz k otevření řádku („open a line“). Znak „[“ pak může být editorem chápán jako šipka. Není-li nastaven aplikační režim („Cursor Key Application Mode“), po stisknu-tí šipky se pošle ESC [D. Obvykle je režim ESC [D implicitní, takže zdánlivě je vše v pořádku. Až na to, že databáze termcap obsahují řetězec (nikoli inicializační), který nastaví terminál tak, jak jste nechtěli: do „aplikačního režimu“. Při spuštění editoru tento řetězec nevadí, avšak nyní nastanou problémy.

Řetězec je v termcapu označen jako „ks“ (v terminfo smkx), což znamená aktivaci funkčních (a podobných) kláves (včetně šipek). Aplikace aktivuje tyto klávesy posláním řetězce „ks“ na terminál. Ať už napsal termcap kdokoli, zvolil řešení, že potřebuje-li aplikační program tyto klávesy aktivovat, budou přepnuty do „aplikačního režimu“, neboť je přepne „aplikace“, což však nechcete.

Linuxová konzola nemá řetězec „ks“, takže na konzole do této pasti nespadnete. V případě jiných terminálů asi budete muset

upravit termcap (resp. terminfo) anebo použít jinou položku termcap. Bude nutno změnit nejen řetězec „ks“, nýbrž i definice toho, co posílají kd, kl, kr a ku. Pak změny nainstalujte pomocí programu tic.
Editor vim (vi iMproved) je možné nastavit tak, aby fungoval korektně s posloupností ESC O D (takže není nutné měnit termcap); viz nápověda ke „kurzorovým klávesám vt100“. Můžete napří-klad spustit gitkeys a stisknout kurzorové klávesy, abyste zjistili, co posílají. V editoru však mohou posílat i něco jiného.

Chyby v bashi

S rozhraním pro čtení řádku jsou v bash shellu problémy. Rozhraní verze bash shellu 2.01 (1998) na „hloupém“ terminále totálně havarovalo. V pozdějších verzích byly chyby odstraněny. Jeden problém však přetrvává. Když některé terminály posílají bajty s bitem nejvyššího řádu nastaveným na 1, bash nebere v úvahu jejich významy, jak jsou definované v položce terminfo. Tuto chybu jsem tvůrcům bashe ohlásil. Ostatní programy (vim, lynx) zpracovávají tyto klávesy korektně.

Problém si můžeme přiblížit tak, že v souboru /etc/inputrc nadefinujeme funkce těchto kláves v bashi. Například terminál Wyse 60 v aplikačním režimu pošle při stisknutí šipek posloupnost D0-D3. Když v tomto smyslu upravíme terminfo, v bash shellu nebudou klávesy fungovat ani po této úpravě. Šipky jsem v souboru /etc/inputrc explicitně definoval takto:

Arrow keys in 8 bit keypad mode: Sends d0-d4. -ap means application. $if term=wy60-25-ap set enable-keypad on “\xd0”: backward-char “\xd1”: forward-char “\xd2”: next-history “\xd3”: previous-history $endif

Je-li už terminál v aplikačním režimu, není nutno zadávat příkaz set enable-keypad on. Tento příkaz pošle terminálu řídicí sekvenci, která se v terminfo jmenuje smkx (v případě terminálu wyse60 je to \E~3 a nastaví šipky tak, že posílají D1-D3). Mnohé aplikace ji posílají automaticky.

Když havaruje příkaz ls při barevném výpisu

Když havaruje emulace barevného terminálu, barvy vypněte. Barvy používají příkazy ls --color a ls --colour. V některých instalacích je barevný výpis ls nastaven implicitně. Zkontrolujte ali-asy v souborech /etc/profile apod. Jak fungují barvy v příkazu ls na konzole, viz kapitola „Pří-klady funkce ls“. Na terminále nastavte monochromatický výpis.

Displej zamrzá (terminál zůstává viset)

Příznakem „visícího“ terminálu je stav, kdy se znaky zadávané na klávesnici neopisují na termi-nále (anebo se opisují, avšak jinak se nic neděje). V takovém případě zadejte ^Q, čímž obnovíte tok dat (pokud byl zastavený). Terminál může také viset, když na něj posíláte binární kombinace nebo když se nestandardně ukončil program, který jej používal.

Pokud k ničemu takovému nedošlo, program může být chybný nebo komunikace s ním je fatál
ně chybná. Když chcete takový program ukončit a obvyklé metody selhávají (některé programy se ukončují stisknutím určitých kláves), zkuste jej zrušit z jiného terminálu pomocí příkazů top nebo kill. Nechce-li se proces ukončit, zkuste to signálem č. 9 odshora (nebo zadejte kill na příkazovém řádku). Po vynuceném ukončení typu „9“ se mohou ztratit některé dočasné soubory nebo může zhavarovat rozhraní. Když ani to nezabere, ukončete shell, čímž se spustí program getty a vypí-še nový přihlašovací prompt. Když selže vše ostatní, zaveďte znovu systém, nebo dokonce vypně-te počítač. Při samotném restartu může zůstat nezměněný obsah hardwarových registrů sériového
portu, což může být příčinou havárie. Nezkušený uživatel také může omylem stisknout klávesy Ctrl+S (^S) (nebo klávesu No Scroll), což má za následek záhadné zmrazení obrazovky (ačkoli to je při softwarovém řízení toku správná funk-ce těchto kláves). Normální komunikaci obnovíte stisknutím Ctrl+-Q (^Q). Je nutno si uvědomit, že vše, co napíšete v průběhu „zmrazení“, se provádí, avšak až do stisknutí ^Q bez odezvy na obra-zovce. V takovém stavu nepište na klávesnici nic, co by mohlo zničit soubory apod. Dobrým důvo-dem pro hardwarové řízení toku dat je právě prevence takových zamrznutí terminálu.

Porucha rozhraní

Sem patří i „zamrzlý displej“ = „visící terminál“ z předchozí kapitoly.

Příznaky a některé způsoby odstranění

Když se displej chová podezřele nebo správně neopisuje zadané znaky (pokud vůbec) anebo se při zadání příkazu nic neděje, může jít o poškozené rozhraní. Většinou nejde o hardwarovou chybu sériového portu, kdy jedinou možností nápravy je vypnout počítač a znovu zavést systém, což může někdy stačit. Někdy dokonce pomůže, když se znovu přihlásíte. Předtím však musíte zrušit shell, který běží na terminále (anebo program getty, je-li v činnosti), což lze provést z jiné-ho terminálu. Jakmile ukončíte getty, je tento program ihned spuštěn znovu a může se stát, že se porucha již neprojeví. Podobně může pomoci vypnutí a zapnutí nebo resetování terminálu.

Možné příčiny poruchy:

Chyba v programu (program se například může chybně domnívat, že pracujete na terminále typu „linux“).
Hardwarová chyba (například i taková, která se jindy neprojeví).
Nesprávná konfigurace (např. chyba v terminfo nebo typ terminálu).

Pokud vše fungovalo správně a chyba se projevila náhle, k poruše rozhraní mohlo dojít vinou nějaké vaší činnosti. Nejčastější příčiny mohou být tři:

Posílání binárních kódů na terminál.
Nestandardní ukončení programu.
Omylem zadaný znak Ctrl+S.

Posílání binárních kódů na terminál

Pošlete-li na terminál určitou řídicí sekvenci nebo řídicí znaky, charakteristika terminálu se změní. Pokud se omylem pokusíte vypsat binární soubor, může obsahovat sekvenci, která převede ter-minál do nedefinovaného režimu anebo jej zcela vyřadí z činnosti. Binární soubory prohlížejte a opravujte pouze pomocí programů, které k tomu jsou určeny, aby se něco takového nestalo. Většina editorů a prohlížečů umí zacházet s binárními soubory a rozhraní nepoškodí. Některé mohou vypsat zprávu, že binární soubory neumějí zpracovávat. Pokud ovšem pošlete na stan-dardní výstup příkazy cat ... nebo cp ... /dev/tty... nebo nějakým programem binární soubor, pravděpodobně poškodíte rozhraní terminálu (pokud ovšem není standardní výstup přesměrován operátorem > apod.).

Rozhraní může poškodit i komunikační program, který pošle na terminál binární data ze vzdále-ného počítače. Buďte připraveni na to, že k takovým věcem dochází z nejrůznějších příčin. Nežá-doucí řídicí kódy mohou dokonce obsahovat i textové soubory. V takovém případě je vždy nutno resetovat terminál. Buďto zadejte jen příkaz reset (nemusí fun-govat) nebo setterm -reset (následovaný pochopitelně znakem return). Možná ani nebudete vidět, co píšete. Tímto příkazem pošlete na terminál obnovovací řetězec, který je uložen v pří-slušné položce databáze terminfo. Jinou možností je obnovení terminálu z nastavení uloženého v paměti terminálu, což se obvykle provádí speciální klávesou (klávesami), a to nejspíš v nasta-vovacím režimu. Pak ještě asi bude potřeba poslat pomocí programu tset na terminál inicializač-ní řetězec.

Příkaz reset nefungoval, chyba je nyní odstraněna

Poznamenejme, že v roce 2000 se příkaz reset jevil jako nefunkční pro terminály, které byly nasta-vovány pomocí programy clocal, neboť pravděpodobně rušil nastavení provedené programem clocal. V roce 2001 byla asi tato závada odstraněna, není tedy nutno číst zbytek tohoto odstavce. Pokud takový problém přece jen máte, příkazem reset situaci jen zhoršíte, neboť přerušíte komu-nikaci mezi terminálem a počítačem. Nejspíš se budete muset znovu přihlásit a doufat, že program getty nastaví terminál pomocí clocal. Vypíše-li se prompt, aniž jste o to požádali, máte štěstí. Jinak je nutno nahlédnout do kapitoly „Příznaky a odstraňování“. Příkaz reset byste měli vyzkoušet ještě předtím, než jej budete skutečně potřebovat, a budete-li po jeho zadání odhlášeni, zkuste příkaz setterm -reset. Chybu jsem ohlásil v březnu r. 2000, avšak oznámení o odstranění jsem nedostal.

Poškozené znaky

Vypisují-li se na obrazovce podivné znaky, zkuste zadat ^O, možná to pomůže. Stejnou službu udělá i příkaz reset, avšak ten obnoví i všechny ostatní funkce terminálu. Může nastat i případ, že všechny znaky vykazují stejnou nežádoucí vlastnost (jsou nejasné, příliš jasné nebo i neviditelné, avšak mezery zadané tabelátorem jsou normální. Tato nežádoucí vlast-nost písma může být způsobena řídicí sekvencí, jež se nevztahuje na tabelátor. Zkuste terminál resetovat apod.

Nestandardní ukončení programu

Rozsáhlé aplikační programy (např. editory) často používají příkaz stty (nebo podobný), aby v průběhu své činnosti dočasně změnily konfiguraci stty. Ovladač zařízení může přejít do „syro-vého“ režimu, kdy se zadávané znaky předávají přímo aplikačnímu programu. Opisování znaků ovladačem je zablokované a znaky na obrazovce vypisuje přímo aplikační program. V takových aplikacích pak nemusí fungovat některé řídicí příkazy (např. ^C).

Když dáte takovému programu pokyn k ukončení, nejdříve vrátí stty do stavu, v němž byl před jeho spuštěním. Pokud program ukončíte nestandardně, může být terminál v „syrovém“ režimu. Když se po ukončení aplikačního programu znaky nevypisují na obrazovce, pravděpodobná pří-čina může spočívat právě v nestandardním ukončení aplikačního programu.

Ze syrového režimu k normálnímu nastavení stty se dostanete příkazem stty sane. Musíte jej však zadat bezprostředně za znakem return a také jej tímto znakem ukončit. Pokud ovšem je terminál ve stavu, kdy se znak return nepřevádí na znak „nový řádek“, což očekává shell, příkaz sane se neprovede. V takovém případě zkuste ukončit příkaz znakem „nový řádek“ (^J) sami přímo z klá-vesnice. Rozhraní vytvořené příkazem sane nemusí být zcela totožné s normálním rozhraním, avšak obvykle funguje. Je-li příčina havárie rozhraní jiná, můžete zkusit také příkaz stty sane.

Speciální (řídicí) znaky

Řídicí znaky zadávané z klávesnice zpracovává ovladač, který pak na jejich základě provádí pří-slušné akce (povely). Vypsat si je můžete příkazem stty -a, poté si prohlédněte 2. – 4. řádek. Jsou podrobně popsány v manuálových stránkách stty. Pomocí programu stty je možné je měnit nebo blokovat, nemusí tedy přesně odpovídat tomu, co je zde uvedeno. Používají se k editaci příkazo-vých řádků, přerušování, rolování obrazovky a k transparentnímu (přímému) zadávání následují-cího znaku.

Editace příkazového řádku

Zatímco ovladač terminálu má několik příkazů na editaci příkazového řádku, některé shelly mají vestavěný svůj vlastní editor (např. „readline“ v bash shellu). Obvykle je implicitně v činnosti, takže není nutno jej aktivovat. Pokud máte takový editor k dispozici, není nutno se zabývat násle-dujícími povely, i když běžně je možné je používat současně s editorem. Nejdůležitějšími povely jsou ^C (přerušení), ^D (konec souboru) a ^S (zastavení rolování obrazovky).

Klávesa delete smaže poslední znak (označuje se někdy jako ^?).
^U zruší celý řádek.
^W zruší předchozí slovo.
^R opíše řádek. Příkaz je užitečný pouze na terminálech, které píší na papír??

Přerušování (& ukončení, odstavení, konec souboru/EOF, zrušení)

^C je přerušení. Ukončí program a vrátí řízení na prompt příkazového řádku.
^\ ukončení. Totéž co přerušení. Taktéž může do souboru v pracovním adresáři vypsat kus paměti (což vám nejspíš k ničemu nebude).
^Z je pozastavení. Zastaví program a odsune jej na pozadí. Jeho činnost obnovíte příkazem fg.
^D je konec souboru. Když jej zadáte za promptem, ukončí se shell a předá se řízení tam, kde bylo před spuštěním shellu.
^O je zrušení. Dosud není implementováno v Linuxu (je v plánu). Výstup pošle na /dev/null.
^T vypíše stav ovladače. Dosud není implementováno v Linuxu (je v plánu). Vypíše sta-vový řádek rozhraní (počet odeslaných bajtů apod.).

Zastavení/obnovení rolování

Když to, co chcete vidět, roluje pryč z obrazovky, můžete rolování zastavit signálem „stop“ (^S nebo Xoff) na počítač (za předpokladu, že je aktivováno řízení toku dat). Rolování obnovíte zasláním signálu „start“ (^Q nebo Xon). Některé terminály mají klávesu No Scroll, která střídavě posílá Xoff a Xon anebo možná i hardwarové signály pro řízení toku dat?? Funkce ctrl-S (^S) a ctrl-Q (^Q):

^S zastav rolování (Xoff),
^Q obnov rolování (Xon).

Chcete-li současně ukončit rolování i program, zadejte ^C. Chcete-li zastavit rolování a dělat něco jiného, avšak program, který vypisuje, ponechat v paměti a později jej obnovit, zadejte ^Z (poza-stavení).

Alternativou k metodě rolování je posílání výstupu do roury prostřednictvím stránkování, např. more, less nebo most. Výstup však nemusí být standardním výstupem a může obsahovat chyby, které stránkování nerozpozná. Odstranit je můžete přesměrováním 2>&1, spravíte tím stránková-ní. Často však je lepší zadávat pouze ^S a ^Q, pokud nepotřebujete rolovat obrazovku v opač-ném směru.
Na konzole PC (na níž je emulován terminál) můžete rolovat zpět pomocí Shift+PageUp. To potřebujete často, neboť rolování je příliš rychlé, než aby šlo snadno zastavovat pomocí ^S. Když budete rolovat zpětně, rolování dopředu pak obnovíte klávesou Shift +PageDown.

Transparentní zadání znaku

Znak následující za povelem ^V (obvykle řídicí) nebude ovladač nijak interpretovat a pošle jej beze změny. Na obrazovce se místo něho vypíší dva ASCII znaky, například ^C.

Zobrazování souborů v Latin1 na non-Latin1 terminálech

Některé „textové“ soubory jsou zakódovány do osmibitového kódu Latin 1 (=ISO-8859-1), viz kapitola „Znakové množiny“. Pokud terminál znaky Latin 1 nezná (anebo nemáte-li nainstalova-nou znakovou sadu Latin 1), některé symboly (např. puntíky) se budou zobrazovat chybně. Při prohlížení manuálových stránek (jsou zakódovány v Latin 1) zadejte volbu -7, aby se znaky pře-vedly do ASCII. Existují stránkové prohlížeče, které tento převod provádějí??

Jak odstranit zvýraznění znaků

Prohlížíte-li manuálové stránky příkazem man a výstup přesměrujete do souboru, bude takový soubor obsahovat „zvýraznění“ znaků, což je vlastně vytváření tučných znaků pomocí dvojího pře-tisku téhož znaku na jedno místo. V souboru jsou takové znaky skutečně uloženy dvakrát za sebou a jsou odděleny znakem „posuv zpět“ („backspace“). Při tisku na tiskárnu to nevadí (i když tisk je obvykle tak tmavý, že přetiskem se nic nevylepší), avšak co s tím, když chcete takový sou-bor poslat elektronickou poštou nebo jej editovat?

Přetisků se můžete zbavit například pomocí příkazu ul (underline, podtržení). Zadáte: ul -t dumb my_overstruck_file > output_file. Příkaz ul převede přetisky na tučné znaky pro daný terminál tak, že do výstupního souboru přidá řídicí sekvence,

které je vygenerují. Jenomže „hloupý“ ter-minál nemá žádné řídicí sekvence, takže tím dosáhnete požadovaného výsledku. Existují i jiné způsoby, tento je však pro „hloupý“ terminál nejvhodnější.

Jak získáme informace o rozhraní

Informace o rozhraní terminálu vám poskytnou tyto nástroje:

gitkeys: bitové kombinace (bajty) posílané jednotlivými klávesami.
tty: port, ke kterému jste připojeni.
set: hodnota vnější proměnné TERM (položka v databázi terminfo).
stty -a: nastavení všech stty.
setserial -g /dev/tty?? (?? si doplňte) vypíše typ UART, adresu portu a číslo IRQ.
infocmp: běžná položka v databázi terminfo (bez komentářů).

Změna nastavení terminálu

Nastavení terminálu je normálně provedeno jednou, když terminál instalujete pomocí procedur setup, jak popisuje manuál terminálu. Nicméně, některá nastavení lze změnit i za provozu. Pří-klady stty (resp. setserial) se při činnosti terminálu obvykle nezadávají, neboť by mohly způso-bit havárii rozhraní terminálu. Existují však změny, které si můžete dovolit provést. Nemají vliv na integritu rozhraní a týkají se zejména vzhledu obrazovky a chování terminálu. Někdy tyto změny provedou samy aplikační programy a nemusíte se o ně starat.
Jednou z přímých metod, jimiž takové změny můžete provést, je nastavovací klávesa (nebo podobná) na terminále a následné použití nabídky (nebo něčeho podobného). Budete-li prová-dět tyto změny, měli byste dobře znát příslušný terminál. Další tři metody spočívají v zasílání tří řídicích sekvencí na terminál. V následujících příkladech je pomocí těchto metod provedeno inverzní zobrazení textu:

setterm -reverse,
tput -rev,
echo ^[[7m.

setterm

Tento příkaz je nejjednodušší. Obsahuje dlouhou volbu (avšak nikoli s obvykle používaným sym-bolem --). Příkaz zjistí v databázi terminfo, jaký má poslat kód. Můžete jím změnit barvu, jas, zala-mování řádků, opakování kláves, vzhled kurzoru apod.

tput

Příkaz tput je podobný příkazu setterm, avšak jako parametry musí mít zkratky používané v data-bázi terminfo, nikoli běžná slova. Mnohé zkratky nevyjadřují význam a nesnadno se pamatují.

echo

V příkladu echo ^[[7m, kterým nastavujete inverzní zobrazování textu, je znak ^[řídicím znakem. Zadáte jej pomocí znaků ^V^[(nebo pomocí znaku ^V následovaného klávesou Escape). K této metodě si musíte v terminálovém manuále nebo v databázích termcap, resp. terminfo, najít pří-slušný kód. Na příkazovém řádku je snazší používat příkazy setterm a tput. Naopak ve skriptech (které se provádějí zejména při spouštění systému) je vhodnější příkaz echo, protože neprohle-dává žádnou databázi, a je tedy rychlejší.

Ukládání změn

Změny, které jste provedli, budou po vypnutí terminálu ztraceny (pokud je ovšem v nastavova-cím režimu neuložíte do permanentní paměti terminálu). Chcete-li je znovu uplatnit, aniž byste je museli znovu opisovat, uložte příslušné příkazy do skriptu nebo si vytvořte shellovou funkci, kte-rou v případě potřeby můžete znovu spustit. Jedním z možných způsobů, jak takové změny mohou být „jako trvalé“, je, když jsou tyto příkazy uloženy v souboru, který se spouští při přihla-šování nebo při zavádění systému.

Vícenásobná sezení

Pomocí balíku screen můžete provozovat vícenásobná sezení, podobně jako virtuální terminály na konzole, viz kapitola „Konzola: /dev/tty?“. Mezi jednotlivými sezeními se můžete přepínat. K podobnému účelu slouží i komerční program Facet Term, viz *http://www.facetcorp.com/ facetterm-overview.html.*

Odhlašování

Odhlásíte se příkazy logout nebo exit. Příkaz může být za jistých okolností odmítnut, avšak důvod vám bude sdělen. Jedním z důvodů odmítnutí může být skutečnost, že pracujete v jiném shellu, než v kterém jste se přihlásili. Odhlásit se také můžete pomocí ^D. Vzhledem k tomu, že povel ^D se používá také k jiným účelům, může vám připadat odhlašování tímto povelem nevhodné. V

takovém případě nastavte v bash shellu IGNOREEOF, povel už nebude sloužit jako odhlášení.

Konverzace mezi terminály, slídění

Když si dva lidé přihlášení na terminálech ke stejnému počítači chtějí povídat, mohou k tomu pou-žít programy write nebo talk (lepší). Příkazem mesg lze zamezit posílání zpráv (psaní) na vlast-ní terminál (s výjimkou superuživatele).

Ke zjišťování toho, co někdo jiný dělá na terminále, použijte program ttysnoop. V tomto progra-mu mají dva terminály stejný status a vše, co napíšete na některé z klávesnic, se objeví na tomtéž místě na obou obrazovkách. Chcete-li tedy někoho sledovat, nepište nic.

Program ttysnoop lze také používat k výuce, kdy žák a vyučující sedí vedle sebe, každý u jed-noho terminálu. Učitel může sledovat, co žák píše, a opravovat chyby tak, že zadává správný text. Žák sleduje učitelův text a může jej opisovat, jak kdyby oba měli ruce na jedné klávesnici sou-časně.

Nevýhodou programu ttysnoop je, že oba terminály musí mít (nebo emulovat) stejný typ termi-nálu (např. vt100). Vzhledem k tomu, že emulace „Linuxu“ na konzole (monitoru) a emulace minicom jsou podobné, ttysnoop je možné používat na dvou PC, přičemž na jednom z nich běží minicom, který emuluje terminál.

Existuje také komerční program DoubleVision podobný ttysnoop, který toho však umí daleko víc. Terminály nemusí být stejného typu, umí si zapamatovat a zopakovat sezení na terminálech, takže můžete zpětně zkoumat průběh sezení. Program naleznete na adrese *http://www.tridia.com.*

Sdílení sériového portu

Jeden sériový port je také možné využívat současně pro textový terminál a pro jiné sériové zaří-zení, např. tiskárnu nebo modem. Pomocí jedné z následujících metod je připojení velmi jedno-duché:

Odpojte kabel od terminálu a zasuňte jej do jiného zařízení.
Mezi terminálem a zařízením přepínejte pomocí AB přepínače (budete k tomu potřebovat 3 kabely).
Port pro tiskárnu (resp. pomocný port) na terminále použijte pro jiné zařízení.

Nepoužíváte-li sériový port pro připojení terminálu, musíte se přesvědčit, zda někdo omylem neposílá na tento port znaky, které chce posílat na terminál. Možným, avšak málo bezpečným způ-sobem, jak to udělat, je nechat běžet programy, které běží na terminálu, a předpokládat, že nepo-sílají na terminál nic, když používáte jiné zařízení. To někdy funguje, neboť terminál na sériovém portu nijak nebrání jinému procesu, aby si tento port otevřel. Takhle to pracuje, když ono druhé zařízení je tiskárna. Jestliže však druhé zařízení posílá nějaké bajty na sériový port počítače, pak program (programy), který dosud běží na portu, bude posílat bajty zpět na terminál, jež však budou směřovat na ono jiné zařízení a ztratí se.

Abychom tomu předešli, je nejlepší ukončit všechny programy (procesy), které běží na terminále, ještě dřív, než začnete používat druhé zařízení. Není to ovšem tak jednoduché, jak se zdá. Na portu obvykle běží shell (např. bash), a když jej ukončíte (například se odhlásíte), program getty se spustí znovu na tomto portu a bude čekat na nové přihlášení. Když ukončíte getty, spustí se znovu. Na první pohled se tedy zdá, že na sériovém portu terminálu není možné ukončit všech-ny procesy.

Jedním ze způsobů, jak lze tento problém obejít, je změnit úroveň běhu tak, aby na portu nebě-žel ani getty, ani shell. V případě pevné úrovně běhu můžete vytvořit jinou úroveň, v níž na portu neběží žádné getty. Pak do této nové úrovně vstoupíte a začnete používat sériový port k něčemu jinému. K tomu musíte upravit soubor /etc/inittab a zjistit (resp. i nastavit) prioritu getty.
Například řádek „S1:23:respawn:/sbin/getty ...“ znamená, že getty poběží (na ttyS1) na úrov-ni 2 a 3. Nyní změníte úroveň pouze na 2 (vynecháte „3“) a jiné sériové zařízení budete používat s úrovní 3. Když tedy budete chtít používat sériový port pro něco jiného, superuživatel zadá init 3, čímž se přepne do úrovně 3. Poznamenejme, že každá úroveň běhu může mít jinou množinu inicializačních skriptů, můžete to však změnit tak, že jeden skript poběží na obou úrovních. Skrip-ty a úrovně běhu jsou závislé na konkrétních distribucích. V případě Debianu naleznete vysvětle-ní v souboru /usr/doc/sysvinit, nahlédněte také do souboru /usr/share/doc.

Existuje také problém s konfigurací portu pomocí stty. Port používaný pro terminál má určitou konfiguraci, avšak když ke změně úrovně běhu použijete řekněme init 3 a na portu zablokujete getty, neobnoví se původní (zaváděcí) konfigurace a pravděpodobně zůstane v „syrovém“ reži-mu. To znamená, že sériový port by nejspíš měl být programem stty plně zrekonfigurován, a to buď spuštěním ze skriptu nebo z aplikačního programu běžícího na portu. Některé aplikace, např. tisk ze sériového portu, úplnou rekonfiguraci neprovedou. Starší verze /etc/printcap zrekonfi-guruje port jenom částečně (avšak novější LPRng už plně). Možná si tedy budete muset napsat vlastní skript. Konfigurace terminálu pomocí stty se budou lišit v závislosti na tom, zda na portu běží shell, zda čeká na přihlášení (vypsal přihlašovací prompt) apod. Konfigurace po změně úrov-ně běhu tedy mohou být různé.

Prohlížeče pro terminál

Prohlížeč lynx pracuje korektně se všemi terminály. Existují i jiné textové prohlížeče: w3m, links a elinks, které jsou spolehlivé pouze na linuxové konzole a terminálech xterm a vt100. Prohlížeč netrik vyžaduje barevný terminál. Prohlížeče links a elinks musí mít nastaven terminál na 8 bitů bez parity (což jsem ohlásil jako chybu, neboť údajně mají pracovat i s paritou). Pro

zakódovanou komunikaci podporují všechny (včetně netrik??) ssl (secure socket layer).

V prohlížečích w3m, links a elinks byly odstraněny některé nedostatky, které jsou v lynx; elinks je pouze vylepšený links. Lépe vypisují tabulky a umí zobrazit rámečky vedle sebe. Bohu-žel však jsou zobrazované údaje (tabulky a rámečky) často širší než obrazovka a přesahují přes její pravý okraj. Obrazovku je nutno posunout ve vodorovném směru, takže zase zmizí z obra-zovky jména řádků. Z toho důvodu je v některých případech lépe použít lynx.

Prohlížeče w3m, links a elinks narozdíl od lynx bohužel nemají číslované řádky a ani nemají náležitou podporu cookies. Žádný z textových prohlížečů zatím nemá podporu Javascriptu, údaj-ně se na nich teprve pracuje (snad v roce 2003). Prohlížeče links a netrik mají podporu Java-scriptu prozatím pouze částečnou.

Existovaly i další projekty textových prohlížečů, avšak některé z nich zřejmě usnuly.

Speciální využití terminálů

Jak udělat z terminálu konzolu

Popíšeme si, jak můžete změnit textový terminál (nebo emulátor PC) na „sériovou konzolu“. Řada systémových zpráv je posílána pouze na konzolu (monitor PC). Když po úspěšném zavedení syste-mu zadáte příkaz dmesg, některé zprávy také mohou být posílány na terminály. To je však k niče-mu, neboť nepodaří-li se zavést systém, nebudou funkční ani terminály. Jádro Linuxu lze ovšem upravit tak, že terminál může sloužit jako konzola a budou na něm vypisovány všechny zprávy smě-rované na konzolu. Bohužel, jediné, co se na terminále nevypíše, jsou zprávy od BIOSu (to je to první, co se vypisuje po zapnutí počítače). Tyto zprávy se vypíší na monitoru v každém případě.
Na toto téma existuje návod, který se jmenuje Remote-Serial-Console-HOWTO. Někdo provozuje PC pouze se sériovou konzolou, tedy bez monitoru a bez klávesnice. Obvykle není možné spus-tit PC bez klávesnice a bez videokarty, avšak některé BIOSy to umožňují. Máte-li to štěstí, že tako-vý BIOS máte (umí přesměrovat konzolu), pravděpodobně bude stačit jen při zapnutí PC nastavit BIOS pomocí nabídky CMOS.

Metoda, jak vytvořit „sériovou konzolu“, závisí na verzi jádra. V každém případě by ovšem měla být sériová podpora součástí jádra, nikoli zaváděným modulem.

Jádra 2.2 a vyšší

Pokyny k vytvoření sériové konzoly jsou uvedeny v dokumentaci jádra (v souboru documenta-tion/serial-console.txt). Dokumentace k jádru 2.4+ je zpracovaná lépe a je v ní zmínka
o nutnosti použít na sériový port program getty. Zařízení /dev/console se obvykle odkazuje na tty0 (konzola PC). Nový soubor /dev/console pro sériovou konzolu vytvoříte příkazem:

```
mknod -m 622 console c 5 1
```

Příkazy pro ovládání sériové konzoly musíte doplnit i do souboru /etc/lilo.conf a spustit jej. Důvodem je skutečnost, že před zavedením jádra je nutno spustit ekvivalent programu setserial, aby mohl být na sériové konzole vypisován průběh zavádění systému. Podrobnosti viz výše zmí-něná dokumentace jádra a také manuálové stránky lilo.conf.

Zde je příklad souboru /etc/lilo.conf (pro ttyS0): prompt timeout=50 boot = /dev/sda vga = normal # force sane state install = /boot/boot.b message = /boot/message image = /vmlinuz root = /dev/sda1 label = console serial = 0,9600n8 append = “console=ttyS0”

Jeden počítač může mít i několik sériových konzol, avšak hlavní konzolou, která komunikuje s počítačem, se stane konzola uvedená v příkaze append jako poslední?? Viz dokumentace jádra (není to z ní ovšem příliš jasné).

Jádra před verzí 2.2

V roce 1997 vyšel v dubnovém čísle časopisu Linux Journal článek o záplatách jádra Linuxu. Nazačátek souboru src/linux/drivers/char/console.c přidáte několik příkazů #define:#define CONFIG_SERIAL_ECHO#define SERIAL_ECHO_PORT 0x2f8 /* Adresa sériového portu */

Následující příkazy už v článku nebyly. V jádru 2.+ (a starších ??) musíte také nastavit rychlost pře-nosu (nevyhovuje-li 9 600). V následujících dvou řádcích:

```
serial_echo_outb(0x00, UART_DLM); /* rychlost 9600 */ serial_echo_outb(0x0c, UART_DLL);
```

zaměníte 0x0c za (vyberete požadovanou přenosovou rychlost): Vybíráte-li zaváděný systém (pomocí LILO) na konzole, avšak chcete pracovat na terminále, musí-te přidat jeden řádek do souboru /etc/lilo.conf. Najděte si v manuálových stránkách soubor lilo.conf a vyhledejte v něm řetězec „serial=“.

115200 baudů: 0x01	19200 baudů: 0x06	2400 baudů: 0x30
57600 baudů: 0x02	9600 baudů: 0x0c	1200 baudů: 0x60
38400 baudů: 0x03	4800 baudů: 0x18	

Provozování Linuxu bez monitoru

Jedním z možných způsobů, jak provozovat Linux bez monitoru, je používat jako monitor sério-vou konzolu. Viz kapitola „Jak udělat z terminálu konzolu“. Přesto můžete potřebovat videokartu, neboť mnohé BIOSy ji vyžadují při spouštění systému. Mohou rovněž vyžadovat klávesnici anebo mají volbu, jíž je možno tuto podmínku v BIOSu zrušit.

Zavádíte-li systém bez monitoru, přesvědčte se, že se zavaděč (např. LILO nebo GRUB) nepokouší poslat na obrazovku žádný obrázek. Textový terminál jej neumí zobrazit a zavádění by mohlo zůstat „viset“.

Máte-li terminál bez klávesnice, můžete si z něho pomocí programu ttysnoop také udělat moni-tor. Zatím však nepracuje jako konzole, neboť nedostává všechny úvodní zprávy o zavádění systé-mu. Viz následující kapitola.

Jak používat terminál bez klávesnice jako monitor Principy činnosti

I když o terminálu bez klávesnice si můžete myslet, že je k ničemu, přece jen jej lze použít jako monitor a psát přímo na klávesnici PC. Lze to provést pomocí „špionážního“ programu ttysnoop. Pomocí tohoto programu můžete na jednom terminále sledovat („čenichat“), co se děje na jiném terminále (nebo i na konzole/monitoru).

Předpokládejte nyní, že píšete na monitoru tty1, a představte si, že vás někdo (pomocí progra-mu ttysnoop) sleduje na monitoru ttyS2. Nyní postavte „špionážní“ terminál (na ttyS2) vedle monitoru tty1. Vše, co píšete na klávesnici PC, se nyní bude zobrazovat na dvou obrazovkách před vámi: na monitoru i na špionážním terminále. Nyní při psaní pozorujte pouze špionážní ter-minál. A teď rozpojte terminál a klávesnici, která je k němu připojena a kterou stejně nepoužívá-te. Od tohoto okamžiku jste začali používat terminál bez klávesnice jako monitor. Jednoduché a vtipné řešení!

Může vzniknou problém (který, jak se ukáže, vlastně ani problémem není), že by špionážní a sle-dovaný terminál asi měly být stejného typu. Vzhledem k tomu, že monitor je obvykle deklarován jako terminál typu „linux“ (hodně podobný vt100), mohlo by se zdát, že skutečný terminál by také měl být (anebo alespoň emulovat) vt100. To však vůbec není nutné! Když například máte termi-nál wyse60, stačí (nesprávně) deklarovat, že na tty1 máte terminál wyse (tj. programu getty řek-nete, že tty1 je wyse60).

Zde je důvod, proč se tímto problémem nemusíte zabývat. Vraťme se ke scénáři v okamžiku, kdy máte před sebou monitor i terminál wyse60 a oba vypisují totéž (anebo se o to pokoušejí). Pro-gram ttysnoop posílá na wyse60 stejné bajty jako na monitor. Pokud jste nesprávně zadali, že monitor je typu wyse60, ttysnoop pošle na monitor i na terminál wyse60 řídicí sekvenci wyse60. Výpis na wyse60 bude v pořádku, zatímco výpis na monitoru bude nesmyslný, neboť obdržel řídi-cí sekvenci wyse60. Vy ale monitor nepoužíváte (a nejspíš jej i odpojíte), nesmyslný výpis tedy vlastně nikomu nevadí (stačí jen dělat, že jej nevidíte).

Příklad konfigurace

Nejde o ideální nastavení, neboť ttysnoop je spuštěn tak pozdě, že se ani nevypíše přihlašovací prompt. Tento příklad se vztahuje k terminálu wyse60 na ttyS2 a k chybějícímu monitoru/PC klá-

vesnici na tty1. Místo getty je zde použit agetty z distribuce Debian. Program getty by mohlvyžadovat jiný formát.V /etc/inittab:

```
1:2345:respawn:/sbin/getty 38400 tty1 wyse60 -ln /usr/sbin/ttysnoops
```

Poznamenejme, že ttysnoop/ttysnoops je kombinace klient/server, takže „s“ na konci není pře-klep. Nezadáte-li -n, možná se vám na terminále nevypíše přihlašovací prompt, neboť agetty jej pošle dřív, než je aktivovaný ttysnoop. S volbou -n neposílá prompt agetty, nýbrž login. Zadá-te-li -n, agetty nebude automaticky zjišťovat paritu, a pokud ji nepoužíváte, vše bude v pořádku.

V /etc/snooptab: # tty snoopdev type execpgm tty1 /dev/ttyS3 init /usr/local/bin/sterm

Pomocí skriptu sterm (shora) se spouští program stty. Nemusí to být nutné anebo může být vhod-ný pro jiný účel. Příklad skriptu sterm v souboru /usr/local/bin/sterm: #!/bin/sh stty rows 24 /bin/login $@

Odstraňování závad

Domníváte-li se, že jde o hardwarovou závadu, viz kapitola „Opravy a diagnostika“. V případě nefunkční klávesnice viz kapitola „Klávesnice“. Opisují-li se nesprávně znaky z klávesnice (pokud se vůbec něco opisuje), viz kapitolu „Poškozené rozhraní terminálu“. Týká-li se problém sériové-ho portu samotného, viz kapitola „Návod k sériovému portu“.

Zde je seznam možných závad:

Podezření na vadný terminál. Je v pořádku?
Displej zamrzá (terminál zůstává viset).
Vypisují se podivné znaky a symboly.
Vypisují se řídicí sekvence.
Chybějící text. Některý text se nevypíše nebo se vypíše, ale zůstává viset.
Všechny klávesy píší chybně nebo je nutno je stisknout několikrát.
Po spuštění programu getty z příkazového řádku se program zastaví a vypíše se „stopped“.
Program getty se spouští příliš rychle (zpráva „chyba konzoly“).

Po přihlášení zhavaruje.
Nelze se přihlásit, avšak přihlašovací prompt je vypsán.
Zkomolený přihlašovací prompt.
Nevypisuje se přihlašovací prompt.

Závady terminálů můžeme rozdělit do dvou skupin. První skupina závad jsou případy, kdy termi-nál už fungoval, avšak najednou nejde. O těchto závadách je následující podkapitola. Druhou sku-pinou jsou případy, kdy terminál nefunguje hned po nainstalování. V takovém případě můžete následující kapitolu přeskočit.

Terminál fungoval

Když terminál fungoval a něco se s ním stane, závadu lze obvykle nalézt poměrně snadno. Vylou-číme-li hardwarovou závadu, problém vznikne většinou v důsledku něčeho, co jste sami způso-bili (nebo co způsobil software, který jste použili).

Problém může být tak očividný jako chyba vypsaná při prvním zapnutí terminálu. Vydá-li podivný zvuk, asi potřebuje opravit. Viz kapitola „Opravy a diagnóza“. Nejdříve přemýšlejte, co se v poslední době změnilo a co nejspíš způsobilo problém. Nastal problém ihned po nainstalování nového systémového softwaru nebo po změně konfigurace?

Neodpovídá-li terminál na zadávaný text, jak má (odpovídá-li vůbec), můžete mít poškozené rozhraní terminálu (viz kapitola „Poškozené rozhraní terminálu“).

Nově nainstalovaný terminál

Pokud jste terminál právě připojili k počítači podle návodu a on nefunguje, tato kapitola je urče-na právě vám. Pokud terminál, který fungoval, nyní nefunguje, viz kapitola „Terminál fungoval“. Máte-li podezření na vadný sériový port, můžete zkusit spustit diagnostický testovací program. V červnu 1998 se zdá, že Linux takový program ještě neměl a budete muset použít diagnostiku DOS/Windows. Existují programy pro monitorování různých sériových linek, jako např. DTR, CTS apod., které mohou pomoci lokalizovat závadu. Viz Monitorování/diagnostika sériového připoje-ní.

Jednou z možností je vyzkoušet, zda bude terminál fungovat, když na něj ve zcela jednoduché situ-aci pošlete kopii souboru (cp my_file /dev/ttyS?). To znamená se zablokovanými řídicími linka-mi modemu a s přenosovou rychlostí odpovídající rychlosti zobrazování na obrazovce, jež nevy-žaduje řízení přenosu (přesvědčte se, zda je řízení toku zablokované). Pokud kopírování funguje, situaci trochu zkomplikujte, zkuste znovu zobrazit kopii souboru atd. Pokud se závada projeví po bezprostředně po určité změně, příčinou bude pravděpodobně právě tato změna. Ve skutečnosti je efektivnější (avšak složitější) přeskočit z jednoduché situace zhruba na poloviční cestu ke koneč-né konfiguraci tak, abyste vyloučili zhruba polovinu možných příčin závady a při příštím testu tuto metodu zopakovali. Z tisíce možných příčin tak naleznete skutečnou příčinu zhruba po deseti tes-tech. Nic se nestane, když tuto metodu půlení intervalů nebudete aplikovat zcela přesně.

Je terminál v pořádku?

Při zapnutí některých terminálů se na obrazovce objeví nějaká slova. Pokud nejde o chybové hlá-šení, nic se neděje. Jestliže do terminálu nejde proud (tmavá obrazovka apod.), zkuste na obou stranách vytáhnout a znovu zasunout síťový kabel. Přesvědčte se, zda je zásuvka, resp. prodlužo-vací šňůra, pod proudem. Máte-li jiný síťový kabel, vyzkoušejte jej. Zkontrolujte, jestli je terminál zapnutý a zda není spálená pojistka. Příčinou světlé (nebo tmavé) obrazovky může být stažený jas nebo kontrast nebo některá klávesa může být v nastavovacím režimu.

U delší dobu nepoužívaného terminálu může trvat zahřívání poněkud déle, než se pod napětím zahřejí elektrolytické kondenzátory. Pokud ani to nepomůže, viz tipy v kapitole „Opravy a diagnóza“.

Máte-li podezření, že se zapnutým terminálem není něco v pořádku, přejděte do „lokálního reži-mu“, v němž funguje jako psací stroj, a zkuste něco napsat. Viz kapitola „Lokální režim“. Než se pustíte do hledání, přesvědčte se, zda terminál umí ohlásit hardwarovou chybu. To můžete zjistit například tak, že zapnete terminál s odpojenou klávesnicí – měl by ohlásit chybu.

Chybějící text

Vypíše-li terminál několik řádků a pak přestane psát (například uprostřed slova apod.) anebo chybí určité úseky textu, vypadá to na problém s řízením toku dat. Pokud nepřijdete na příčinu ihned, zkuste snížit přenosovou rychlost. Pomůže-li to, jste nejspíš na správné stopě. Řízení nemu-sí fungovat vůbec vinou nesprávné konfigurace nebo kvůli vadné kabeláži (v případě hardwaro-vého řízení). Viz kapitola „Řízení toku dat!“.

Pokud funguje psaní na klávesnici, avšak z každých poslaných 16 znaků z počítače se jich vypí-še jen několik (nebo i jen jeden), pravděpodobně jste zadali programu setserial nesprávný UART. Stává se to u starších portů (16550 a nižších), zatímco programu jste zadali 16550A nebo vyšší.

Chybí-li v textu jen jednotlivé znaky, pravděpodobně je zahlcen sériový port příliš vysokou pře
nosovou rychlostí. Zkuste ji snížit.Máte-li zařízení s přenosovou rychlostí nižší než 1 200 baudů (například starý terminál, který píšena papír nebo tiskárny) a text je uříznutý, příčinou může být ovladač sériového zařízení. Viz návod„Tisk v Linuxu“, kapitola

„Sériové tiskárny“.

Klávesy šifrují nebo je nutno je stisknout několikrát

V této situaci musíte klávesu stisknout několikrát, než zafunguje (a dívat se, jestli se skutečně vypsala). Napíšete-li slovo, mohou chybět některá (nebo všechna) písmena. Chybí-li písmena v příkazu, neprovede se. Také se může stát, že písmena v příkaze jsou sice všechna, avšak musí-te opakovaně stisknout klávesu return, než se příkaz provede.

Příčinou může být skutečnost, že sériový port si otevřely dva různé procesy a oba se snaží číst znaky z klávesnice. Jeden znak přečte jeden proces (ten správný, například shell), další znak pře-čte druhý proces. Může to být třeba proces, který obsluhuje sériovou myš (např. gpm) a který pře-čtené znaky neopisuje. Takže zadávané znaky „žere“ jiný proces, který běží na stejném ttySx. Které procesy běží na ttySx, zjistíte pomocí příkazu ps -alx a ten, který způsobuje závadu, může-te ukončit.

Mohli byste spoléhat na to, že podobným situacím předejdete pomocí programu lockfiles. Avšak ani terminál, ani program gpm na ovládání myši program lockfile nepoužívají, a to právě z toho důvodu, aby na terminál mohl psát i někdo jiný.

... příliš rychlé spouštění: zablokovat na 5 minut Co se děje

Na konzole uvidíte například zprávu „Getty respawning too fast: disabled for 5 minutes“. Místo „getty“ tam může být návěští (např. Id „S2“), kde S2 je návěští řádku v souboru /etc/inittab, odkud byl volán program getty.

Když se spustí getty, zkusí poslat na sériový port přihlašovací zprávu. Je-li však v systému něco závažnějšího v nepořádku, getty je okamžitě ukončen. Vzhledem k tomu, že řádek, kterým se spouští getty v souboru /etc/inittab, říká, že má být spuštěn znovu, getty se skutečně spustí a znovu je ukončen, což se opakuje stále dokola. V této situaci zasáhne systém a toto nesmyslné počínání (na pět minut) ukončí. V následující kapitole jsou uvedeny pravděpodobné příčiny a jejich odstraňování.

Nesprávný řádek getty v souboru /etc/inittab

Přesvědčte se, že řádek, který v souboru /etc/inittab volá getty, je zadán správně. Problém může způsobit i překlep v „ttySx“ (v programu uugetty je to„DTxxxx“) nebo v „getty“.

Chybí řídicí signál od modemu

Neposílá-li terminál signál CD na jeden z pinů sériového portu PC, program getty bude ukončen, neboť v programu getty nebyla zadána volba „local“. Problém snadno odstraníte tím, že zadáte tuto volbu (v programu agetty -L, v programu ps_getty v souboru /etc/gettydefs „CLOCAL“).

Jinou možností je zjistit, proč není nastaven CD. Často se stává, že k pinu CD nevede žádný vodič, takže musíte zadat volbu „local“. Pokud je pin CD zapojený, nemusí na něm být signál, dokud nezapnete terminál síťovým vypínačem. Poznamenejme ještě, že někdy je na terminále (anebo i na počítači) pin CD nahrazen pinem DTR a jsou propojeny v konektoru.

Takové sériové zařízení neexistuje

Je-li zařízení jen hypotetické (fyzicky neexistuje nebo je zablokované), getty bude ukončen. Zaříze-ní můžete vyzkoušet programy scanport (pouze v Debianu??) nebo setserial, nebo můžete zkusit jiné zařízení. Zařízení bylo pravděpodobně zablokováno CMOS, viz návod „Serial-HOWTO“.

Chybí podpora sériového připojení

Podporu sériového připojení buď musíte zadat při překladu jádra nebo zavedením sériového modulu v době běhu: serial.o. K chybě „opakovaného spouštění“ programu getty dochází tehdy, když nebyl proveden žádný z těchto úkonů. Zda je sériový modul zaveden, můžete zjistit pomocí příkazu lsmod. Pokud byste chtěli zjistit, zda je zabudovaný v jádru, budete si muset pro-hlédnout konfigurační soubor jádra (v /boot, ve zdrojovém stromě apod.).

Zkratovaná klávesa

Další možnou příčinou častého opakovaného spouštění programu getty může být zkratovaná klá-vesa, což se projevuje stejně, jako kdybyste drželi klávesu trvale stisknutou. Je-li aktivováno opa-kování klávesy, jako odpověď na prompt generuje klávesa tisíce znaků. Podívejte se, zda není celá obrazovka popsána stejnými znaky (v některých případech i dvěma nebo více různými znaky).

Zhavaruje ihned po přihlášení

Pokud se úspěšně přihlásíte, avšak ihned poté je terminál nefunkční, důvodem může být skuteč-nost, že shell po spuštění zrekonfiguruje terminál (nastaví jej nesprávně) příkazem, který je obsa-žen v některém ze souborů, které se provedou po přihlášení a po spuštění shellu. Patří k nim sou-bory /etc/profile a ~/.bashrc. Vyhledejte příkaz, který začíná na stty nebo setserial, a pře-svědčte se, zda v něm není chyba. To však ještě nemusí stačit. I když jeden inicializační soubor funguje správně, může nastavení změnit jiný inicializační soubor, o kterém nevíte. V takovém pří-padě je nutno opravit systém pomocí záchranné diskety z jiného terminálu nebo z konzoly, pří-padně na lilo prompt odpovědět příkazem „linux single“. Po jeho provedení přejde systém

do jed-nouživatelského režimu, ve kterém se neprovádějí startovací soubory.

Není možné se přihlásit

Vypíše-li se přihlašovací prompt, avšak na pokusy o přihlášení počítač neodpovídá (anebo odpo-vídá zmateně), pravděpodobnou příčinou je nesprávná komunikace ve směru z terminálu na počí-tač. Může to být špatný nebo nesprávně propojený kabel, resp. konektor. Pokud jste ještě nepře-šli do lokálního režimu, přejděte do něj, abyste vyřadili z činnosti řídicí linky modemu. Viz kapi-tola „Getty (použití v /etc/inittab)“. Můžete také zablokovat hardwarové řízení toku dat (stty -crt-scts), je-li aktivováno. Pokud se terminál po těchto změnách rozběhne, pravděpodobně nejsou správně zapojeny řídicí linky modemu anebo je chyba v nastavení. Některé terminály mohou mít nastavenu jinou hodnotu (např. přenosovou rychlost) na vysílání a jinou na příjem, takže příjem je v pořádku, avšak vysílání je nastaveno chybně. Také byste mohli zkusit zadat na konzole příkaz stty < /dev/ttyS1 (používáte-li ttyS1), abyste viděli, že je nastaven korektně. Často bývá v „syrovém“ režimu (což je pravděpodobně správné) s volbami -icanon nebo -echo apod. Je-li terminál nastaven chybně v poloduplexu (HDX), jedna sadu znaků, kterou vidíte při psaní, jde přímo z terminálu. Jsou-li znaky zdvojeny, ozvěny z počí-tače jsou v pořádku a terminál můžete přepnout do plného duplexu. Je-li však nastaven polodu-plex a vidíte pouze zdánlivé „ozvěny“, ve skutečnosti nejdou z počítače, jak by správně měly.

Dostanete-li zprávu „login failed“ (nebo podobnou) a neudělali jste chybu ani v přihlašovacím jménu, ani v hesle, mohli jste být odmítnuti kvůli určitým omezením, jimž podléhá přihlašování. Bohužel, zpráva nic neříká o důvodu odmítnutí. Viz kapitola „Omezení přihlášení“.

Nesprávný přihlašovací prompt

Může být způsoben nesprávnou znakovou množinou, chybami přenosu při vysoké rychlosti, nekompatibilních přenosových rychlostech, nesouhlasné paritě nebo chybném počtu bitů v bajtu. Je-li chybných znaků mnoho, používáte nesprávnou znakovou množinu nebo je nejvyšší bit omy-lem nastaven na jedničku. Jsou-li ve slovech chyby, zkuste snížit přenosovou rychlost. K chybám souhrnně označovaným jako „chybný znak“ dochází vinou nekompatibility přenosové rychlosti, parity nebo počtu bitů/znak a takto poškozený znak je vypsán zcela jinak než znak původní, nepoškozený. Může vypadat třeba jako obrácený otazník, obdélník nebo jinak podivně.

Program agetty (často se jmenuje jen getty) při psaní rozpozná a nastaví paritu a počet bitů na znak. Můžete si to vyzkoušet, řádek ukončete znakem return.

Chybí přihlašovací prompt

Pokud se ani po opakovaném stisknutí klávesy return nevypíše přihlašovací prompt, proveďte následující kontrolu: Pomocí programů top nebo ps se přesvědčte, zda skutečně na terminálu pro-bíhá proces getty (nebo login). Je terminál připojen ke zdroji napětí? Je nullmodemový kabel ve správných portech jak na terminále, tak i v počítači?

Také zjistěte, zda proces getty „nevisí“, tj. nečeká na signál CD (a zda je vůbec v kabelu přísluš-ný vodič napojený na signál CD). Pokud není signál přítomen, musíte některým ze dvou násle-dujících způsobů zadat programu getty volbu „local“:

V programu getty_ps (Redhat apod.) jsou to dva příznaky CLOCAL v souboru /etc/get-tydefs (viz kapitola „getty – getty_ps“).
V programu agetty (Debian apod.) příznak -L v souboru /etc/inittab.

Pokud neprobíhá proces getty (resp. login), pečlivě zkontrolujte tvar volání getty v souboru /etc/inittab. Přesvědčte se, zda obsahuje správnou úroveň běhu (zjistíte ji příkazem runlevel) a zda je příslušná k vaší verzi getty. Někdy pomůže zrušení procesu getty, resp. login (automa-ticky se spustí znovu).

Terminál pracoval správně

I když nelze vyloučit ani hardwarovou chybu, pravděpodobnější bude chyba lidského faktoru. Nemohl někdo poškodit kabel? Neupravoval někdo soubor /etc/inittab nebo neprováděl jiný zásah do přihlašovací procedury? Pokud ne, čtěte dál.

Další diagnóza

Postupy shora by měly vést k odhalení nejčastějších příčin (pokud jste si právě nenainstalovali ter-minál). Dalšími příčinami může být chyba v hardwaru nebo v kabeláži (kabel musí být určen k přenášení souborů), nastavení terminálu na nesprávnou rychlost přenosu, terminál může být v lokálním režimu atd. V tomto bodě se vydáme dvěma směry (můžete se jimi vydat současně).

Diagnostika problému z konzoly.
Měření napětí.

Diagnostika problému z konzoly

Jedním z testů je zkusit něco zkopírovat na terminál (vhodný způsob zejména krátce po startu instalační procedury, ještě než nastavíte getty). Můžete zadat třeba příkaz cp file_name /dev/ttyS1 anebo echo any_word > /dev/ttySx. Pokud nefungují, co nejvíce zjednodušte roz-hraní pomocí programu getty tak, že zablokujete co nejvíce funkcí (tj. hardwarové řízení toku dat:

-crtscts; paritu a signály modemu: clocal). Přesvědčte se, zda jsou shodné přenosové rychlosti a počty bitů/znak. Když nic z toho nepomůže, ověřte pomocí voltmetru, zda je port „živý“, viz následující kapitola.

Měření napětí

Máte-li voltmetr, zkontrolujte na souborovém kabelu na straně terminálu na pinu 3 (receive data, příchozí data) záporné napětí (od -4 do -15 V). Kladnou zdířku voltmetru dobře uzemněte (kovo-vé konektory na kabelu často nebývají dostatečně uzemněny). Pokud na pinu nenaměříte zápor-né napětí, změřte ještě přenosový pin (TxD) na počítači (rozložení pinů viz kapitola „DB9-DB25“). Pokud tam záporná hodnota je, avšak není na pinu pro příjem (RxD) na terminále, kabel je vadný (špatné propojení, přerušený vodič anebo to není souborový kabel zvaný též nullmodem). Není--li napětí na počítači, je vadný port. Otestujte jej diagnostickým programem anebo jej rovnou vyměňte.

Je-li sériový port živý, můžete na něj zkusit poslat soubor (se zablokovaným řízením modemu) a sledujte signál na voltmetru (nebo na jiném elektronickém přístroji). Na voltmetru sledujte, zda je napětí trvale záporné, když se nic nepřenáší. Pak začněte posílat soubor (nebo spusťte getty). Na analogovém voltmetru by při přenosu bitů měla ručička klesnout téměř k nule a chvět se kolem ní. Na digitálním voltmetru změny neuvidíte, avšak přepněte na střídavé napětí, které je vlastně vytvářeno tokem bitů. Pokud voltmetr nelze přepnout na střídavé napětí (u většiny analo-gových voltmetrů implicitní), můžete improvizovaně sledovat střídavé napětí na stejnosměrné stupnici na -12 V. Nemáte-li voltmetr, můžete k sériovému portu připojit zařízení, o němž víte, že je funkční (např. jiný terminál nebo externí modem), a vyzkoušet je.

Vypisuje podivné znaky a symboly

Nezaměnujte za kapitolu „Vypisuje řídicí posloupnosti“. Jestli se na obrazovce vypisuje něco jiného, než co byste očekávali, avšak vypadá to jako nějaká cizí abeceda, matematické symboly, kreslené znaky apod., může být, že většina bajtů posílaných na terminál má nastavený nejvyšší bit (i když by neměla mít). Vypisuje se znaková množina (nebo její část) s nejvyšším bitem rovným
1. To se může stát, když máte špatně nastavenou přenosovou rychlost nebo paritu (na stty). Máte--li paritu nastavenou, avšak v terminále jsou osmibitové znaky bez parity, nejvyšší bit (=paritní) bude často nastaven chybně. Zkuste zadat z jiného terminálu stty -F /dev/ttyS?, abyste ověřili správnou přenosovou rychlost a paritu.
Také je možné, že byla nainstalovaná nesprávná znaková množina (fonty). Chybná řídicí sekven-ce poslaná na terminál může přepnout znakové množiny. Pokud používáte ke změně mapování kláves program mapchan, může být chyba v něm.

Vypisuje řídicí sekvence

Na obrazovce se objevuje něco jako „5;35H22,1“ nebo „3;4v“ nebo „1;24r“ nebo „^[[21;6H“ atd. Pochopitelně čísla a písmena mohou být jiná. Budou přeházená (buď náhodně nebo v nezvyklém pořadí). Na displeji bude změť znaků a terminál bude pravděpodobně vykazovat i jiné závady. Některé aplikace a příkazy budou vypisovat nesmysly.

To, co vidíte na displeji, jsou řídicí sekvence (nebo jejich zlomky), které byly poslány na terminál, aby jej řídily, avšak terminál je nerozpoznal a poslal je na obrazovku. Pravděpodobně používáte program, který se chybně domnívá, že používáte jiný terminál. Proto posílá řídicí sekvence, kte-rým terminál nerozumí. Na displeji se mohou dít podivné věci. Ověřte, zda je správně nastavena vnější proměnná TERM (zadejte: echo $TERM).

Telnet

Problém, jak získat správnou hodnotu proměnné TERM, může být v případě používání telnetu poněkud složitější. Telnet neemuluje terminál, nýbrž předává hodnotu proměnné TERM vzdálené-mu počítači. Pokud ten nepodporuje používaný typ terminálu anebo na vzdáleném počítači chyb-ně nastaví hodnotu TERM, nastanou potíže. Telnet by měl nejdříve na vzdáleném počítači správ-ně nastavit hodnotu TERM. Změny hodnoty proměnné TERM na vzdáleném počítači mohou být způsobeny nesprávným konfiguračním souborem shellu. Nejdříve musíte zkontrolovat hodnotu proměnné TERM jak na vašem, tak i na vzdáleném počítači. Výklad je poněkud zjednodušený, neboť je možné, že váš telnetový klient vám ohledně serveru poskytne celý seznam hodnot pro-měnné TERM, které váš počítač podporuje (pokud telnet ví, že váš počítač umí emulovat víc než jeden typ terminálu).

Terminál je nastaven tak, aby vypisoval řídicí sekvence

Jinou možnou příčinou může být skutečnost, že terminál je ve zvláštním režimu, v němž řídicí sekvence neprovádí, nýbrž vypisuje. V takovém případě je rovněž uvidíte na obrazovce, avšak budou vypsány uspořádaně. Přesně řečeno je to režim pro výpis řídicích kódů. Vzhledem k tomu, že všechny řídicí sekvence začínají řídicím kódem (znakem „escape“), terminál nepozná, že jde
o řídicí sekvenci, a celou ji pošle na obrazovku. Viz kapitola „Řídicí kódy“.

Zpomalení: znaky jsou vypisovány s

několikasekundovým zpožděním

Pravděpodobně máte špatně nastaveno přerušení – viz Serial-HOWTO, kapitola, která začíná slovem „Slow“.

Terminál neroluje

Příčinou může být chyba v databázi terminfo. Anebo je možné, že jste mimo rolovací oblast termi-nálu. Některé špatné programy předpokládají, že terminál má 24 řádků, a na tuto hodnotu nastaví (řídicí sekvencí) rolovací oblast, aniž by se podívaly do terminfo na skutečný počet řádků. Jiný pro-gram pak umístí kurzor třeba na řádek 25, kde zůstane trčet a terminál neroluje. Tento problém odstraníte tak, že vytvoříte a nastavíte vnější proměnnou: export LINES=25 a také stty -F /dev/ttySx rows 25. Pak si snad program, který počítá s 24 řádky, nastaví rolovací oblast správně.

Monitorování sériového portu/diagnostika

Některé linuxové programy mohou monitorovat řídicí linky modemu a sdělovat, jestli jsou kladné (1) nebo záporné (0).

statserial (distribuce Debian).
„Soubor“: /proc/tty/driver/serial. Monitorování zadáte pomocí „watch head...“. Poky-tuje informace o chybách a datovém toku.
Modemstat (pracuje pouze na linuxové konzole PC; lze jej zadat na příkazovém řádku).

Možná, že tyto programy už máte. Pokud ne, naleznete je na stránkách Serial Software (*http://ibiblio.unc.edu/pub/Linux/system/serial/*). Při používání mějte na paměti, že tyto programy signalizují stav linek na centrálním počítači. Situace na terminále může být jiná, neboť některé vodiče velmi často chybějí a neodpovídá jejich propojení. K červnu 1998 nevím o žádném dia-gnostickém programu pro sériový port.

Lokální režim

V lokálním režimu je terminál odpojen od počítače a chová se jako psací stroj (s tím rozdílem, že nepíše na papír, nýbrž na obrazovku). Když se vrátíte do on-line režimu, terminál se znovu při-pojí k počítači a můžete pokračovat tam, kde jste skončili při přechodu do „lokálního“ režimu. Této možnosti lze využít jak pro testování terminálu, tak i pro výuku. Některé terminály lokální režim nemají, nahrazuje jej však „blokový režim“. Pokud ani ten nemají, může stačit „poloviční duplex“ s tím rozdílem, že text zadávaný na klávesnici je přenášen do počítače. V takovém pří-padě může počítač znaky opakovat, což by se projevilo zdvojeným výpisem všech znaků zada-ných na klávesnici. Tento jev odstraníte tak, že vypnete počítač, odpojíte kabel RS-232 apod.

V lokálním režimu můžete zadávat řídicí sekvence (uvedené klávesou ESC) a sledovat, jak na ně terminál reaguje. Nepracuje-li terminál správně v lokálním režimu, je nepravděpodobné, že bude fungovat, když jej připojíte k počítači. Nejste-li si zcela jisti, jakou má určitá řídicí sekvence funk-ci, můžete si ji vyzkoušet v lokálním režimu. Také si můžete takto vyzkoušet terminál, když jej kupujete. Některé terminály při přechodu do lokálního režimu vyžadují, abyste je nejdříve pře-pnuli do nastavovacího režimu a režim „local“ vybrali z vypsané nabídky (anebo stiskli určitou klá-vesu). Viz kapitola „Přechod do nastavovacího (konfiguračního) režimu“.

Měřicí přístroje
Přístroje pro testování linek atd.

Zatímco v případě testování několika sériových portů si vystačíte s multimetrem (zapojeným jako voltmetr), pro testování sériových linek byly vyvinuty speciální jednoduché přístroje. Mají několik konektorů, které se připojují ke konektorům sériového portu (jak na straně kabelu, tak i na zadní straně počítače). Některé mají měřicí body pro připojení voltmetru, jiné mají diody LED, které se rozsvítí, když je na řídicí lince signál. Polaritu signálu (kladné nebo záporné napětí) pak mohou indikovat i barevně. Jiné přístroje mají možnost vodiče propojovat a přerušovat.

V roce 2002 prodávala firma Radio Shack pod katalogovým číslem 276-1401 testovací přístroj „RS--232 Troubleshooter“ (původně nazývaný „RS-232 Line Tester“). Zelená kontrolka znamená +12 V (zapnuto), červená -12 V (vypnuto). Také nabízejí přístroj „RS-232 Serial Jumper Box“, kat. číslo 276-1403, jehož pomocí propojíte libovolné piny. Obě položky najdete v katalogu pod ozna-čením „Peripheral hookup helpers“. Bohužel nejsou v rejstříku tištěného katalogu. Jsou na téže straně jako konektory typu D, takže v rejstříku je nutno je hledat pod označením „Connectors, Computer, D-Sub“. Mohou je také mít obchodní řetězce, které prodávají „aktivní komponenty“.

Měření napětí

Běžné voltmetry nebo multimetry, i ty nejlevnější, které stojí kolem 10 dolarů, by měly úplně sta-čit. Jiné metody zjišťování napětí jsou nepřímé. Diodu LED nepoužívejte, pokud ji nemáte zapo-jenou v sérii s odporem, který sníží proud protékající diodou. Pro 20 mA diodu (existují i diody s jinou hodnotou) použijte odpor 470 Ω. Dioda bude svítit pouze při určité polaritě, takže lze zkoušet jen kladné nebo záporné napětí. Nezkoušeli jste si zhotovit podobný přístroj pro testová-ní elektrických obvodů

ve výbavě automobilu?? Mohli byste si poškodit sondu, neboť napětí logic-kých obvodů, pro něž je určena, je pouze 5 voltů. Zkoušet dvanáctivoltovou žárovkou není dobrý nápad, neboť ta jednak neukáže polaritu a navíc kvůli omezenému výstupnímu proudu UART se pravděpodobně ani nerozsvítí.

Při měření napětí na samičím konektoru můžete do otvoru vsunout sponku na papír. Průměr drátu, z něhož je zhotovena, by neměl být větší než pin, abyste nezničili kontakt. Na sponku pak připojte krokodýlka (nebo něco podobného). Dbejte, abyste se nedotkli žádným kovovým předmětem dvou pinů současně.

Ochutnávka

Když nemáte žádné testovací zařízení a nevadí vám rána elektrickým proudem (anebo že vás to zabije), jako poslední možnost můžete zkoušet napětí jazykem. Než se pokusíte otestovat někte-rý vodič, vyzkoušejte, zda v něm není vysoké napětí. Dotkněte se jednou rukou dvou vodičů sou-časně, abyste viděli, jestli kopou. Pokud nikoli, olízněte si ruku v místě dotyku a zkuste to znovu. Dostanete-li ránu, určitě nezkoušejte vodiče jazykem.

12 voltů otestujete tak, že si olíznete prst a dotknete se jím vodiče. Druhý vodič si dejte na jazyk. Je-li vodič na jazyce kladný, ucítíte výraznou chuť. Nejdříve si to můžete vyzkoušet na baterii do kapesní svítilny, abyste věděli, jakou chuť můžete očekávat.

Oprava a diagnóza

Oprava terminálu má mnoho společného s opravou monitoru a klávesnice. Někdy je na obrazov-ku vypsána vestavěná diagnostika. Závadu lze často lokalizovat podle příznaků: vadná klávesni-ce, nefunguje obrazovka, chyba v elektronice (zdeformovaný obraz) nebo závada v digitálním systému. Nejlepší je mít servisní manuál, avšak jde to i bez něho.

Knihy a internetové stránky určené pro opravy Knihy

Bigelow, Stephen J.: Troubleshooting & Repairing Computer Monitors, 2. vyd., McGraw-Hill, 1997. Nezahrnuje elektroniku pro generování znaků a ani klávesnici.

Internetové stránky

Často kladené otázky (FAQ), *http://www.repairfaq.org*, diskusních skupin: skupina sci.electro-nics.repair je rozsáhlá a má široký záběr, i když není určena jenom pro terminály. Viz kapitola „Computer and Video Monitors“. Řadu informací z jednotlivých částí lze využít i pro terminály: „Testing Capacitors“, „Testing Flyback Transformers“ atd. Možná v budoucnosti budou informace v tomto návodu obsahovat pouze odkazy na často kladené otázky uvedené výše (anebo podob-né). Dalším zdrojem informací je archiv diskusní skupiny *Shuford's repair archive* (*http://www.cs.utk.edu/~shuford/terminal/repair_hints_news.txt*).

Bezpečnost

V barevných obrazovkách bývá napětí až 30 000 voltů (v monochromatických poněkud nižší). Je--li monitor pod napětím a má odstraněný zadní kryt, musíte být velice opatrní a ničeho se nedo-týkat. Pravděpodobně by vás to nezabilo, protože protékající proud je nízký. Avšak nejspíš byste byli popálení, utrpěli šok apod. U vysokého napětí dochází k výboji, který může projít porušenou izolací, takže ruce mějte v bezpečné vzdálenosti. K jedné straně obrazovky je připojen důkladně zaizolovaný kabel. I když je přístroj vypnutý, je v místě připojení kabelu k obrazovce tak velké zbytkové napětí, že by vám mohlo dát elektrickou ránu. Toto napětí můžete vybít šroubovákem (s izolovanou rukojetí) s kovovým ostřím uzemněným k zemnícímu vodiči kabelu. Neuzemňujte jej ke kostře!

Nižší napětí (stovky voltů) mohou být nebezpečnější, neboť není omezen protékající proud. Nebezpečí se zvyšuje, když máte vlhké ruce nebo hodinky s kovovým páskem, prsten apod. Stane se, že někoho to i zabije, takže buďte velice opatrní! Nižší napětí s hodnotami pouhých několik voltů v číselných obvodech jsou zcela bezpečná, avšak nedotýkejte se ani jich (leda dobře izolovanými nástroji), pokud si nejste zcela jisti.

Nastavení displeje

Je-li displej matný, pomocí ovládacích prvků umístěných zvenku na zařízení (pokud tam jsou) zvyšte jas a kontrast. Displej má ovládací prvky také na šířku, výšku a vycentrování obrazu. U některých starších terminálů se nastavování provádí šipkami.

Někdy je nutno kvůli nastavení odstranit z displeje zadní kryt, zejména u starších modelů. Když nastavujete displej, postavte si před něj větší zrcadlo tak, abyste v něm displej viděli. Nastavovací prvky mohou být na tištěných obvodech desky. Možná budete potřebovat jen obyčejný nebo kří-žový šroubovák, na cívky se používají speciální nástroje (plastové pinzety). Na desce s obvody by měla být uvedena zkratka druhu nastavení. Zde jsou příklady těchto zkratek:

V-Size: nastavení výšky obrazu.
H-Size: nastavení šířky obrazu. Může to být cívka.
V-Pos: umístění obrazu ve svislém směru.
H-Pos: umístění obrazu ve vodorovném směru.
V-Lin: nastavení svislé linearity (když řádky v horní a dolní části obrazovky nejsou stejně silné).

V-Hold: nastavení vertikální stability (když obrazovka nekontrolovaně ubíhá).
Nastavení jasu (knoflík může být ovládaný i zvenku).
Pomocné nastavení jasu ve ztlumeném režimu (často jde o normální režim: je matnější než běžný jas).

Změna linearity může změnit velikost obrazu, kterou je pak nutno znovu nastavit. Terminál, který se po určitou dobu nepoužíval, může mít zmenšený obraz lemovaný tmavým okrajem. Než jej začnete nastavovat, nechte jej chvíli zahřát, obraz se může trochu zlepšit sám (tmavé okraje se zúží).

Stanovení diagnózy Terminál vydává zvuky nebo se z něho kouří

Vydával-li terminál nějaké zvuky těsně před tím, než se porouchal (anebo po zapnutí, když už je v poruše), mohou tyto zvuky být vodítkem k nalezení závady. Slyšíte-li zvuky nebo vidíte (cítíte) kouř, ihned terminál vypněte, abyste nezpůsobili další škody. Prasknutí může způsobit explodující kondenzátor nebo spálená tavná pojistka. Jiskření se projevuje prskavým zvukem. Problém může být ve vysokonapěťovém napájení (tisíce voltů). Odstraňte zadní kryt. Prohlédněte si kondenzátory, zda nejsou zabarvené, vyduté nebo jinak poškozené. Není-li místo závady viditelné na první pohled, terminál znovu krátce zapněte a sle-dujte, zda odněkud nejde kouř nebo neuvidíte jiskření. Snáze je najdete v tmavší místnosti. Jiskře-ní může způsobit porušená izolace vysokonapěťového kabelu (který vede od vysokonapěťového transformátoru k boční straně obrazovky). V takovém případě je nutno jej natřít izolačním nátě-rem nebo speciální izolační páskou určenou pro napětí řekněme 10 000 voltů.

Vysokonapěťový transformátor může při poruše vydávat pouze jemné cvakání nebo praskání, které je někdy slyšet až po vypnutí a opětovném zapnutí terminálu. Zvuk můžete vystopovat pomocí gumové hadičky (používají se například v automobilech), kterou použijete jako stetoskop. V průběhu hledání se však terminál může poškodit ještě víc, takže si s hledáním pospěšte (avšak se vší opatrností, abyste nepřišli k úrazu elektrickým proudem).

Příčinou spálení tavné pojistky může být zkrat v napájení a její výměnou nemusí být problém odstraněn – může se spálit v důsledku stejného zkratu. Pokuste se nalézt tmavé skvrny způsobe-né vysokou teplotou a součástky, které je mohly způsobit, přezkoušejte. Přepálení pojistky mohou způsobit také zkratované výkonové tranzistory. Lze je proměřit testerem nebo také jen ohmmet-rem. Ohmmetrem začněte měřit na nejnižší stupnici, neboť jí odpovídá i nejnižší zkušební napětí. Minimalizujete tím případné škody na dobrých součástkách, které by toto testovací napětí mohlo způsobit.

Pokud byl terminál po určitou dobu vystaven vlhkosti, např. byl uložen na vlhkém místě, poblíž kuchyně, z níž šla pára, může se podařit odstranit závadu vysušením jednotky. Někdy pomůže, když „vadný“ vysokonapěťový transformátor vysušíte proudem horkého vzduchu (stačí pár minut).

Terminál nevydává žádný zvuk

Není-li na obrazovce vůbec nic, může to být způsobeno nastavením jasu na nejnižší hodnotu, případně stářím obrazovky. Dále je nutno zkontrolovat, zda nemají kabely utržené nebo nalomené konektory. Nejde-li do terminálu proud, zkontrolujte napětí v zásuvce a pak vyzkoušejte jinou síťovou šňůru.

Máte-li podezření na klávesnici, přezkoušejte ji na jiném terminále stejného typu nebo ji nahraďte dobrou klávesnicí. Na obou koncích zahýbejte kabelem klávesnice. Vodiče v kabelu mohou být nalomené, zejména těsně u konců. Pokud při hýbání kabelem zjistíte, že závada mizí a znovu se objevuje, je nutno kabel vyměnit, opravit přerušený vodič apod.

Při zjišťování závad klávesnice nejprve přepněte terminál do lokálního režimu. Pokud v tomto reži-mu funguje, problém bude nejspíš v připojení k počítači (nebo v nesprávném rozhraní), případ-ně v elektronice terminálu.

Detektivní práce

Pečlivě si prohlédněte schéma, snáze naleznete příčinu závady. Zkuste najít změnu zabarvení, prasklinky apod. Občasnou závadu můžete najít i tak, že na součástky poťukáte tužkou (nikoli kovovou, pochopitelně). Přerušený kondenzátor na desce s tištěným obvodem můžete někdy najít tak, že desku ohnete. Zkuste přepájet spoj, kde cín vytvořil kapku nebo kde je ho málo. Pájením můžete polovodič (a jiné součástky) přehřát a zničit, při pájení jej pokud možno ochlazujte. Při odstraňování závady může dojít k poškození jiné součástky, takže i když závadu najdete, může se objevit jiná.
Máte-li běžný typ terminálu, může se vám podařit najít na Internetu příčiny nejčastějších závad ter-minálu (anebo i diskusní skupinu, která se touto problematikou zabývá) a rady k jejich odstraně-ní. Zjistíte-li, která součástka je vadná (např. R214 wyse), můžete se pokusit ji na Internetu najít, možná i s komentářem někoho, kdo měl stejný problém. V tomto komentáři třeba budou zmíně-ny i jiné součástky, které se poškodily současně. Je-li součástka zničena tak, že nelze přečíst, co je na ní napsáno, můžete si ji najít na Internetu. Výrobce může mít i on-line údaje, které nenajdete běžnými vyhledávacími programy.

Chcete-li zjistit, zda funguje digitální elektronika, zkuste (musíte mít funkční klávesnici) na termi-nále s poruchou něco napsat. Tento text se pokuste přečíst a pomocí příkazu copy nebo termi-nálovým komunikačním programem (např. minicom) vypsat na fungujícím terminále (nebo na konzole). Aby se řádek odeslal, možná budete muset na terminále stisknout klávesu return. Někdo se může z jiného terminálu zeptat vadného terminálu na identitu apod. Ukáže se, zda fun-guje obousměrná komunikace.

Chybové zprávy na obrazovce

Uvidíte-li na obrazovce výpis chybové zprávy, jste šťastný člověk. Stává se to při prvním zapnutí terminálu.

Chyba klávesnice

Obvykle to znamená, že není připojena klávesnice nebo že není navázáno spojení. Vážnější problémy viz kapitola „Klávesnice“.

Chyba kontrolního součtu v permanentní paměti (NVR)

Permanentní paměť označujeme NVR („Non-Volatile RAM“). Znamená to, že jsou poškozena nasta-vovací data. Terminál pravděpodobně bude fungovat, avšak je ztracena naposledy uložená konfi-gurace. Zkuste jej zkonfigurovat znovu a konfiguraci uložit, může se to podařit. Na velmi starých terminálech (počátek 80. let) byla paměť CMOS, do níž byla ukládána konfigurace, napájená bate-rií, takže v takovém případě je vybitá baterie. Někdy, po mnohonásobném uložení, dojde k poško-zení mikroprocesoru EEPROM (který nepotřebuje baterii). Chyba paměti se hledá těžko. Pokud se vám to nepodaří, musíte vzít zavděk implicitní konfigurací nebo můžete po každém zapnutí posí-lat na terminál určité řídicí sekvence, jimiž se pokusíte terminál zkonfigurovat.

Kondenzátory

Elektrolytické kondenzátory mají kovový obal, a když se dlouho nepoužívají, může klesat jejich kapacita anebo úplně přestanou fungovat. Někdy stačí terminál na chvíli vypnout. Pokud máte tu možnost, uskladněné terminály třeba jednou za rok zapněte do sítě.

Poznamenejme, že levné kondenzátory, které jsou určeny do audiozařízení, nemusí vydržet náročný provoz ve vysokofrekvenčních obvodech. Ty by měly být osazeny nízkoodporovými (low ESR) kondenzátory. Nepolarizované (NP) kondenzátory buď vyměňte nebo nahraďte bipolárními.

Trvá-li zahřívání displeje několik minut, důvodem mohou být právě vadné elektrolytické kondenzátory. Vadný kondenzátor najdete pomocí jednoduchého triku: Podezřelý kondenzátor zdvojíte funkčním kondenzátorem (musí být určen nejméně pro stejné napětí a musí mít přibližně stejnou kapacitu). Pokud se funkce displeje výrazně zlepší, pravděpodobně jste našli vadný kondenzátor. Při takových pokusech buďte velmi opatrní, pozor na úraz elektrickým proudem! Skutečné napětí proti zemi může být mnohem vyšší než napětí uvedené na kondenzátoru.

Klávesnice Zaměnitelnost

Klávesnice terminálů nejsou stejné jako klávesnice PC. Rozdíl nespočívá pouze v rozložení kláves, nýbrž i v kódech generovaných jednotlivými klávesami. Klávesnice různých výrobců a modelů tedy nejsou zaměnitelné. Někdy může být klávesnice nekompatibilní pouze částečně. Znakové klávesy fungují správně, speciální klávesy však mají jinou funkci.

Princip činnosti

Když stisknete klávesu, ve většině klávesnic se pouze spojí dva vodiče. Převod na kód, který je odeslán do kabelu, provede elektronika v mikroprocesoru klávesnice. Aby nemusely být všechny klávesy propojeny s mikroprocesorem samostatným vodičem (drátem), používá se následující schéma: Vodiče očíslujeme např. od 1 do 10 a od A do J. Vodič 3 vede k několika klávesám, vodič B také vede k několika klávesám, avšak oba vodiče 3 a B vedou současně pouze k jedné kláve-se. Při stisknutí klávesy jsou zkratovány tyto dva vodiče, z čehož mikroprocesor pozná, která klá-vesa byla stisknuta. Toto schéma snižuje počet vodičů (a také počet pinů na mikroprocesoru). Schéma je podobné křížovému vypínači.

Nové a staré klávesnice

I když staré i současné klávesnice vypadají přibližně stejně, princip činnosti je jiný. Staré kláves-nice měly pod každou klávesou spínač uzavřený v pevném plastovém pouzdře. Současné kláves-nice mají pod klávesami velké plastové membrány ve velikosti klávesnice s dírkami. Jsou vloženy mezi jiné dvě membrány, které obsahují tištěné spoje (včetně kontaktů). Když stisknete některou klávesu, stisknou se v určitém bodě i dvě membrány s tištěnými spoji a v tomto bodě dojde ke kontaktu.

Jedním stisknutím napíšete dva různé znaky

Jsou-li v důsledku závady zkratovány vodiče 3 a 4, po stisknutí klávesy 3-B je zkratována i 4-B a mikroprocesor si myslí, že byly stisknuty obě klávesy, tedy 3-B a 4-B. Vypíší se 2 znaky, i když jste chtěli vypsat pouze jeden.

Klávesnice vůbec nefunguje

Pokud nefunguje ani jedna klávesa, zkuste jinou klávesnici (máte-li), abyste si ověřili, že problém je v klávesnici. Jednou z příčin může být přerušený drát v kabelu spojujícím klávesnici s terminá-lem. Nejpravděpodobnějším místem, kde je vodič přerušen, jsou oba konce kabelu. Zkuste jimi při psaní na klávesnici pohybovat, abyste zjistili, zda závada občas nezmizí. Najdete-li vadné místo, můžete kabel v tomto místě opatrně naříznout nožíkem a přerušený vodič spojit. Kabel pak zai-zolujte izolační páskou, zalepte nebo utěsněte. Nefunguje-li klávesnice proto, že je vlhká, je nutno ji nechat vyschnout.

Stisknete b, vypíše se bb atd. (zdvojování znaků)

Pokud jsou zdvojovány všechny znaky, příčina pravděpodobně nebude v klávesnici. Nejspíš je špatně nastavený terminál na poloviční duplex (HDX nebo lokálně echo=on) a všechny znaky jsou opisovány jak terminálem, tak i počítačem. Nejsou-li oba znaky stejné, příčinou může být zkratovaná klávesnice, viz kapitola „Jedním stisknutím napíšete dva různé znaky“.

Opakují se řádky stejných znaků

To se může stát, je-li aktivováno opakování znaků a klávesa je trvale stisknutá (nebo téměř trvale). Klávesa může být zaseknutá nebo zkratovaná – výsledek je stejný.

Klávesa je zaseknutá

Nejdříve ji zkuste několikrát stisknout, což ovšem asi nepomůže. Dále z ní můžete zkusit sejmout klobouček (pokud je snímatelný) a kolíček klávesy trochu prostříknout čisticím prostředkem. Zkuste jím pohybovat střídavě nahoru a dolů a do stran. Pokud to nepomůže, vyjměte celý spí-nač a vyčistěte jednotlivé součástky.

Pokud se rozhodnete odstranit klobouček, viz kapitola „Klávesnice s jednotlivými přepínači“. Opa-kovaně stlačujte kolíček, dokud klávesa nezačne fungovat a vypisovat znaky na obrazovce. Ně-které klávesy se zaseknou kvůli tomu, že mají zespodu ulepený klobouček. Pokud zůstane ve zcela spodní poloze, může být příčinou tohoto problému. I tato místa je nutno někdy vyčistit.

Kolíčkem kývejte do stran malým šroubováčkem, abyste jím současně mohli prstem pohybovat nahoru a dolů. Můžete jím kývat do čtyř stran, vyzkoušejte je všechny. Těmito pohyby vlastně z kolíčku odstraňujete cizorodé částečky, které se nalepily na kolíček z boku a upaplaly ho. Problém se může časem opakovat.

Klávesu vždy vyzkoušejte ihned po opravě a potom po chvilce ještě jednou. Pomalu ji stlačujte a pozorujte, jestli lepí. Při stlačování jí také jemně zakývejte. Při prudším stisknutí si nemusíte všim-nout, že lepí. Můžete tak přijít na klávesu, která sice funguje, avšak v budoucnosti by s ní mohla být potíž.

Zkratovaná klávesa

Máte-li podezření na zkrat v klávese, zkuste nejdříve vyčistit kontakty, viz kapitola „Čištění kontaktů klávesnice“.

Klávesnici jste něčím polili

Jestliže se vám podařilo vylít na klávesnici vodu nebo podobnou tekutinu (anebo byla-li vystave-na dešti, husté mlze nebo jiné vlhkosti), některé (nebo i všechny) klávesy nemusí fungovat. Vlh-kost může klávesu zkratovat (jako kdyby byla trvale stisknutá), a je-li aktivováno opakování klá-ves, může se celá obrazovka popsat jedním znakem. Jestliže jste klávesnici částečně (nebo úplně) vysušili, některé klávesy ještě mohly zůstat nefunkční vinou nečistot, které zůstaly na povrchu kontaktů. U nových klávesnic můžete z klávesnice vyjmout plastové membrány, usušit je a vyčis-tit. Starší typy klávesnic můžete vysušit na sluníčku nebo v troubě (pochopitelně při nízké teplo-tě). Po vysušení mohou některé klávesy ještě vyžadovat vyčištění, viz dále.

Čištění kontaktů v klávesnici

U novějších klávesnic lze plastové membrány snadno vyjmout, prohlédnout je, a je-li to nutné, také vyčistit. Musíte jen povolit několik šroubků, abyste klávesnici rozebrali a dostali se k nim. U někte-rých starších klávesnic IBM nejdou membrány vyjmout bez vylomení několika plastových plošek, které pak musíte nalepit zpět (klávesnice je pro opravy velmi nevhodná). Takovou klávesnici pak můžete opravit jedině tak, že ji ohýbáte, kroutíte jí a bušíte do ní v místech, kde jsou membrány.

Klávesnice se spínači

Následující popis platí pro starší klávesnice, které mají pro každou klávesu samostatný spínač. Než se pustíte do práce s čištěním elektrických kontaktů, zkuste nejdříve obrátit klávesnici vzhůru nohama a ťuknout na vadné klávesy. Mohou z nich vypadnout nečistoty, zvláště když klávesu stisknete silou a prudce, aby zavibrovala. Často pomůže, když stisknutou klávesou kýváte do stran.

Pokud to nepomůže, můžete zkusit vyčistit spínač tekutým čističem kontaktů (dostanete jej v elek-troprodejnách), obvykle ve spreji. Ke spínači se dostanete tak, že nejdříve odstraníte klobouček klávesy (čtvereček, do něhož ťukáte prsty při psaní). Upozornění: U novějších klávesnic nejsou kloboučky snímatelné. Můžete si pomoci šroubováčkem, který zasunete pod klobouček, a prstem přitom zmírňujete příliš velký náklon. Existuje speciální nástroj na snímání kloboučků z kláves, avšak určitě si poradíte i bez něho. Klobouček můžete při snímání trochu naklonit a zaviklat s ním. Může vám najednou i vyskočit a spadnout na zem.

Pak si při čištění kontaktů můžete vybrat ze dvou možností: Buď jej můžete vyčistit přímo sprejem, který nastříkáte na spínač shora, nebo klávesu rozeberete a vyčistíte (nejlepší je, když jde rozebrat snadno). Jinou možností je, že celý spínač vyměníte za nový nebo za použitý, avšak s tím bývá hodně práce (a nový spínač není zadarmo).

Postříkat spínač sprejem bez rozebírání je nejrychlejší způsob, avšak čisticí prostředek se nemusí dostat až ke kontaktům, které má vyčistit. Spínač před ošetřením trochu očistěte. Je-li klávesnice zapojená (anebo je-li napojená na ohmmetr), čisticí prostředek nanášejte pomocí trubičky, která je přiložena ke spreji, aby se čisticí prostředek dostal dovnitř. Při sprejování pohybujte kolíčkem

nahoru a dolů. Dbejte na to, aby se čisticí prostředek nedostal pod sousední klávesy, kde by se mohl smísit s prachem a společně s ním prosáknout do spínačů. Dopustíte-li se této chyby, možná opravíte vadnou klávesu, avšak poškodíte sousední. Pokud se to přece jen stane, ihned opakovaně tiskněte sousední klávesy, dokud nezačnou znovu fungovat.

Klávesnici trochu nakloňte, aby čisticí prostředek lépe zatekl do kontaktů. V případě terminálu CIT101e s klávesnicí Apls to znamená nadzvednout horní řadu číselných kláves. Jak budete při ošetřování kontaktů se spínači mírně pohybovat tužkou nebo šroubováčkem, dejte pozor, aby se čisticí prostředek nedostal na holou kůži (anebo si raději nasaďte rukavice). Klávesnici pak otoč-te vzhůru nohama a přebytečný roztok nechte odtéct. Čím víc roztoku do klávesnice nastříkáte, tím spíš se vám podaří vadnou klávesu opravit, avšak tím víc také můžete znečistit a poškodit sou-sední klávesy, takže musíte pracovat s citem. Jakmile začne klávesa fungovat, trochu ji proťukejte a po několika minutách ji vyzkoušejte znovu, abyste se přesvědčili, že stav je trvalý.

Někdy se stane, že klávesa sice funguje, když je roztok na kontaktech, avšak jakmile za pár minut vyschne, zůstanou na kontaktech nečistoty po odpařeném roztoku, které brání dokonalému kon-taktu. Klávesa pak může znovu začít dělat chyby (pokud vůbec bude fungovat). Proto je dobré ťukat na klávesu, už když vysychá. Některé spínače mají kontakty v plastovém pouzdru téměř zatavené, takže se k nim nedostane skoro žádný čisticí roztok a to málo, co k nim proteče, může s sebou přinést nečistoty (z toho důvodu je dobré očistit horní část spínače, ještě než začnete vstři-kovat čisticí roztok).

Pokud byste chtěli spínač rozebrat, musíte nejdříve zjistit, zda je to vůbec možné (a pokud ano, jak se rozebírá). Někdy lze odstranit kryt spínače, aniž byste jej museli vyjmout z klávesnice. Nej-dříve musíte vypáčit (nebo vytáhnout) vrchní část spínače, která je zajištěna plastovými destička-mi, ty musíte nejdřív odstranit. Možná se vám obojí podaří pomocí dvou malých šroubováků. Ne-smíte páčit silou, mohli byste destičky zlomit. Nejdou-li uvolnit, bude lepší spínač odpájet a vyjmout, pak jej můžete jak opravit, tak i vyměnit. Když je spínač rozebraný, možná ještě kon-takty neuvidíte, jsou těsně u sebe (téměř se dotýkají). Čisticí roztok můžete vstřiknout do štěrbiny mezi kontakty, aby mezi ně zatekl. Potom kontakty tenkým šroubovákem trochu rozevřete, roz-tok se k nim dostane rychleji. Kontakty pak šroubovákem střídavě rozevírejte a nechte vracet do původní polohy, rychleji se vyčistí. Současně pozorujte obrazovku, jestli už klávesa funguje.

Kontakty mohou být zajištěny svorkou, kterou je nutno odstranit. Nepoztrácejte malé součástky, mohou ze spínače vyskočit a už je nikdy nenajdete. Jako poslední možnost můžete zkusit pohyb-livou část spínače, na niž tlačí kolíček, trochu přihnout, aby tlačil na kontakt větší silou. Na mém terminále vypadá tato součástka jako elektrický kontakt, avšak pouze stlačuje skutečný elektrický kontakt přes tenkou izolaci.

Když pak dáváte spínač dohromady, přesvědčte se, zda je pružinka na správném místě. Jde-li po smontování klávesa příliš lehce nebo příliš ztuha, pružinka asi není správně umístěna. Má-li pružinka být zapuštěna do otvoru v kolíčku klávesy, můžete ji tam dočasně „vlepit“ pomocí kapičky vody, kterou kápnete na konec pružinky. Tento konec pak vsuňte do otvoru v kolíčku a spínač rychle složte, než voda vyschne. Pružinka by tak neměla vypadnout. Místo lepení pružinky vodou můžete postupovat také tak, že klávesnici postavíte na výšku nebo vzhůru nohama, aby pružinka při montáži nevypadla.

Obecně

Seznam terminálových příkazů v Linuxu Posílání příkazu na terminál

setterm: dlouhé volby,
tput: zkrácené volby,
tset: pouze inicializace,
clear: smazání obrazovky,
setterm -reset: poslání resetovacího řetězce.

Konfigurace ovladače terminálu

setserial
stty

Terminfo

Terminfo Compiler (tic): překladač terminfo.
toe: seznam terminálů, pro něž existují soubory v terminfo.
infocmp: porovnává nebo zobrazuje položky v terminfo.

Ostatní

gitkeys: přehled bajtů posílaných jednotlivými klávesami na počítač.
tty: přehled připojených tty portů.
set (nebo tset -q): hodnota proměnné TERM, jméno položky v terminfo.
tset: interaktivní nastavení a inicializace TERM.

Internet a knihy Informace o terminálech na Internetu

Shuford's Website (*http://www.cs.utk.edu/~shuford/terminal_index.html*), University of Tennessee, mnoho užitečných informací o textových terminálech.
Informace o terminálech VT, historie: *http://www.vt100.net/.*
Boundless (*http://www.boundless.com/Text_Terminals/*) odkoupil obchod s terminály VT a Dorio od firmy DEC. Specifikace naleznete na odkazech na ADDS, VT a DORIO, pak vyberte odkaz „data link". Odtud pak vyberte odkaz „Go to Specs".
Wyse má podrobné informace (např. řídicí sekvence) ve své znalostní databázi (*http://www.wyse.com/service/support/kbase/wyseterm.asp*). Není tak obsáhlá jako manuál, neboť obsahuje pouze obecné vlastnosti. Současné modely viz *http://www.wyse.com/ products/gpt/index.htm.*
Teemworld Escape Sequences (*http://www.ecc400.com/java/twproae.htm*) je seznam řídi-cích sekvencí (a řídicích znaků) pro některé emulace terminálů (např. VT 100, 300, 420 a Wyse).
ncurses FAQ (*http://dickey.his.com/ncurses/ncurses.faq.html*).
comp.terminals je diskusní skupina o terminálech.

Knihy o terminálech

Sériový port EIA-232 viz knihy o EIA-232 (RS-232) na adrese *http://tldp.org/HOWTO/ Text-Terminal-HOWTO-22.html#RS232_books.*
Opravy viz knihy o opravách & internetové stránky na adrese *http://tldp.org/HOWTO/ Te-xt-Terminal-HOWTO-19.html#repair_info.*
Databáze Terminfo viz dokumenty o databázi Termcap na adrese *http://tldp.org/ HOW-TO/Text-Terminal-HOWTO-15.html#termcap_docs.*

Souhrnné knihy o terminálech

Pokud vím, neexistuje vyhovující souhrnná kniha o textových terminálech. I když tento návod vyšel jako kniha, nepředpokládám, že byste si ji koupili, máte-li přístup k on-line verzi, kterou zhruba každý měsíc aktualizuji. Následující tituly mají spíše historickou cenu:

Handbook of Interactive Computer Terminals, Duane E. Sharp; Reston Publishing Co. 1977 (velmi zastaralá).
Communicating with Display Terminals, Roger K. deBry; McGraw-Hill 1985 (zejména

o synchronních terminálech IBM).

První uvedený titul podává stručný přehled o více než 100 různých modelech zastaralých termi-nálů vyrobených počátkem 70. let minulého století více než 60 různými výrobci. Obsahuje popis principů činnosti, bohužel v diagramu znázorňujícím obrazovku je chybně znázorněno vychylo-vání elektronových paprsků (str. 36). Ve skutečnosti terminály využívají k vychýlení toku elektro-nů magnetismus (dokonce už v 70. letech). V knize je vysvětlena řada moderních technických poznatků, jako např. „vektorové skenování" nebo „princip průniků barev".

Druhá kniha se na rozdíl od prvního titulu se při popisu terminálů nezabývá fyzikálními a elekt-rotechnickými podrobnostmi. Celá jedna kapitola je věnovaná dvojkové soustavě (což je v knize
o terminálech zbytečné, neboť tyto informace jsou běžně dostupné i jinde). V knize jsou popsány zejména terminály IBM (hlavně 3270) v blokových a synchronních režimech. Pro současné ANSI terminály používané v unixových systémech je kniha k nepotřebě. I když je o nich v knize zmín-ka, chybí jakákoli schémata připojení k sériovým portům.

Knihy, které obsahují jednotlivé kapitoly o terminálech

V těchto kapitolách není o terminálech jako takových a o jejich funkcích prakticky nic. Popisují spíše nastavování počítačů (a ovladačů terminálů) v souvislosti s terminály. Vzhledem k rozdílům mezi unixovými systémy jich ani pro Linux většina není vhodná.

- Unix Power Tools, Jerry Peck et. al., O'Reilly 1998. Kap. 5: Setting Up Your Terminal, Kap. 41: Terminal and Serial Line Settings, Kap. 42: Problems With Terminals.
- Advanced Programming in the Unix Environment, W. Richard Stevens, Addison-Wesley, 1993. Kap. 11: Terminal I/O, Ch. 19: Pseudo Terminals.
- Essential System Administration, Aleen Frisch, 2. vyd., O'Reilly, 1998. Kap. 11: Terminals and Modems.

V prvním titulu najdete 3 krátké kapitoly o textových terminálech. Obsahují méně popisů než tento návod, avšak více příkladů.V kapitole 11 v druhé knize je pouze popsán ovladač jako součást systému a jsou vysvětleny parametry příkazy stty nutné pro konfiguraci terminálu.

Třetí kniha je napsána dobře, je věnována více terminálům než modemům.

Nelinuxové operační systémy

V operačním systému MS DOS existuje příkaz „ctty COM2“, který slouží k výpisu příkazového řádku na sériovém terminále (v našem případě COM2). Pak už bohužel není možno používat monitor, neboť MS DOS není víceuživatelský operační systém. Také nelze připojit více než jeden terminál. Tato funkce je tedy prakticky k ničemu. Pokud emulujete DOS pod Linuxem pomocí dosemu, můžete údajně používat několik terminálů. Avšak, také údajně, s nimi nespolupracuje emulace PCTerm (Platí to ještě ??).

Zatímco firma Microsoft nevytvořila „víceuživatelský DOS“, našli se tací, kteří jej vytvořili. Lze v nich používat několik terminálů připojených k jednomu PC. Je softwarově kompatibilní s MS DOS. Jeden z těchto víceuživatelských systémů se jmenuje „REAL/32“, k emulaci terminálů se pou-žívá „pcterm“. Existuje i nastavovací režim „scan“ (scancodes), je nutno jej ovšem nastavit. Ostat-ní operační systémy jako PICK, PC-MOS a Concurrent DOS byly (resp. dosud jsou) víceuživatel-ské a podporují terminály.

V Linuxu existují 3 programy, jejichž prostřednictvím můžete provozovat windowsové aplikace pod Linuxem zdarma: Wine, komerční: VMware a NeTraverse. Mohou používat textové terminály pod DOSem? Systém Wine nikoli, nemá totiž režim DOS. Zbývající dva vyžadují spuštění operač-ního systému MS Windows jako „hostujícího operačního systému“. MS Windows má režim DOS, avšak není víceuživatelský, takže je prakticky nepoužitelný.

V jiných operačních systémech typu Unix je konfigurace počítače pro terminály obvykle podstat-ně jiná než v Linuxu. Zde jsou odkazy na on-line manuály unixových systémů:

SCO's OpenServer Adding Serial Terminals in SCO OpenServer Handbook (*http://osr5doc.ca.caldera.com: 457/HANDBOOK/CONTENTS.html*).
Hewlett-Packard's HP-UX „Configuring Terminals and Modems“ (Url nefunguje.).

Terminologie řídicích sekvencí v příkazech

Anglicky se jmenují *„escape sequences“* nebo „*control sequences“*. Tato příloha je neúplná (a vzhle-dem k obrovskému počtu řídicích sekvencí nemůže být nikdy úplná). Uvádím ji pouze jako pře-hled, pravděpodobně by spíše patřila do nějakého programátorského návodu.

Jako příklad standardní ANSI řídicí sekvence může posloužit posloupnost ESC[5B, která posune kurzor o pět řádků dolů. ESC je znak Escape. Součástí sekvence je parametr 5. Kdyby tam bylo 7, kurzor by se posunul dolů o 7 řádků atd. Slovnímu popisu „posuň kurzor dolů o x řádků“ příka-zu ESC[xB jistě každý rozumí. Horší je však žargon, například „požadavek na terciární atribut zaří-zení“ („tertiary device attribute request“). V této kapitole se pokusím vysvětlit alespoň něco z tajem-né hantýrky, která se používá v příkazech pro zadávání řídicích sekvencí. Kompletní seznam (včet-ně kódů řídicích sekvencí ve standardu ANSI) je uveden v návrhu projektu. Vzhledem k tomu, že řada řídicích sekvencí má stejnou funkci jako volby nastavení, nebudu je zde uvádět.

Seznam řídicích sekvencí

Seznam mnoha (nikoli však všech) řídicích sekvencí pro různé terminály najdete v Teemworld Escape Sequences (*http://www.ecc400.com/java/twproae.htm*). Používají se pro emulaci terminálů a nejsou vždy shodné s odpovídajícími skutečnými terminály. Seznam sekvencí pro terminály VT (není udržován) naleznete na Emulators FAQ (*http://www.cs.ruu.nl/wais/html/na-dir/emulators-faq/part3.html*), vyhledejte „VT“. Manuál si můžete stáhnout z VT Manuals na adrese *http://www.vt100.net/*.

8bitové řídicí kódy

Hexadecimální tabulka řídicích kódů. Fungují na VT2xx a novějších. Nejobecnější je CSI.

ZKRATKA CELÉ JMÉNO HEXADEC. NAHRAZUJE IND Index (o řádek dolů) 84 ESC D NEL Nový řádek 85 ESC E RI Reverse Index (o řádek nahoru) 8D ESC M SS2 Posun o jedno 2 8E ESC N SS3 Posun o jedno 3 8F ESC O DCS Řídicí řetězec 90 ESC P CSI Úvod řídicího řetězce 9B ESC [ST Ukončení řetězce 9C ESC \

Řídicí sekvence pro tiskárnu

Automatický tisk zapnout/vypnout: Když je zapnutý, data z počítače jsou posílána i na ter-minál (a vypisují se i na obrazovce terminálu).
Řízení tisku zapnout/vypnout: Je-li zapnuto, data z počítače jsou posílána pouze na tiskárnu (nikoli na obrazovku terminálu).

Hlášení

Tyto sekvence jsou obvykle požadavkem odeslaným z počítače, jímž žádá o hlášení z terminálu. Terminál odpoví tak, že pošle hlášení (ve skutečnosti jinou řídicí sekvenci) na počítač, které ve svém těle obsahuje určité hodnoty, jež vyjadřují stav terminálu. V některých případech je počítači zaslána zpráva, i když o ni nepožádá, obvykle po ukončení nastavení terminálu. Implicitně se žádná nevyžádaná hlášení neposílají.

Žádost o stav (hlášení operačního stavu): VT100 odpovídá buď „jsem v pořádku“ nebo „nejsem v pořádku“.

Žádost o atributy zařízení: „Zařízením“ je obvykle tiskárna. Je připojena tiskárna? Je při-pravena k tisku?
Požadavek na terciární atribut zařízení (VT): Odpovědí je hlášení zadané v průběhu nasta-vování. Terciární zařízení je třetí zařízení (tiskárna nebo port ??). Prvním zařízením by mohl být počítač, druhým terminál.
Žádost o parametry terminálu: Jaká je parita, rychlost přenosu, velikost bajtu atd. Tento požadavek příliš nedává smysl, protože kdyby počítač tyto údaje už neznal, nemohl by s terminálem komunikovat ani posílat odpovědi.

Pohyby kurzoru

Kurzor je tam, kde se zobrazí další znak přijatý z počítače. Většina pohybů kurzoru je zřejmá. „Indexovat kurzor“ znamená posunout kurzor o jeden řádek níž. Pohyby kurzoru mohou být relativní vzhledem k jeho poloze, např. „posuň o 4 místa doleva“, nebo absolutní, např. „umísti na řádek 3, sloupec 39“. Absolutnímu pohybu říkáme „přímé umisťování kurzoru“ nebo „přímé adresování kurzoru“.

Základní poloha kurzoru je na prvním řádku v prvním sloupci (počátek indexování je 1). Kde však je tato základní poloha na obrazovce, není zcela jasné. Je-li nastaven režim „počátek kurzoru“ = „relativní počátek“, základní poloha je vlevo na horním okraji rolované oblasti (což nemusí být na horním okraji obrazovky). Je-li nastaven režim „absolutní počátek“ (což je totéž, jako když zruší-te oba režimy shora), pak základní poloha je v horním levém rohu obrazovky. Na některých star-ších terminálech režim „počátek kurzoru“ znamená, že je relativní.

Stránky (definice)

Stránkování je vysvětleno v části Stránky. Existuje řada řídicích sekvencí, jež souvisejí se stránka-mi. Text může být kopírován z jedné stránky do druhé a kurzorem můžete pohybovat z jedné stránky na druhou. Přepínání stránek může být automatické: Když je stránka zaplněna (strana 1), další data z počítače jdou na stranu 2. Kurzor může být v jednom okamžiku pouze na jedné stra-ně a znak poslaný na terminál jde na tuto stranu. Pokud není zobrazena, terminál nový text zpra-cuje a uloží jej do paměti, v níž je zobrazovaný text, avšak neuvidíte to (dokud není terminál pře-pnut na tuto stránku).

Sériové komunikace na EIA-232 (RS-232)

Úvod do sériové komunikace

(Většina textu z této kapitoly je nyní v Serial-HOWTO) Textové terminály v unixových systémech (a na PC) jsou obvykle připojeny k asynchronnímu sériovému portu 232 počítače. Obvykle to je RS-232-C, EIA-232-D nebo EIA-232-E. Všechny tři jsou téměř totožné. Původní prefix RS byl nahra-zen EIA (Electronics Industries Association) a později EIA/TIA, když EIA fúzovala s TIA (Tele-communications Industries Association). Port EIA-232 spec se také používá pro synchronní (sync) komunikaci, avšak osobní počítače většinou nemají hardwarovou výbavu pro synchronní komu-nikaci. Označení RS je už poněkud zastaralé, avšak dosud se používá. Zde budeme používat EIA.

Sériový port je něčím víc než pouhým fyzickým konektorem na zadní straně počítače nebo ter-minálu. Je vybaven elektronikou, která generuje signály v souladu se specifikací EIA-232. Stan-dardní konektor má 25 pinů, z nichž většina není použita. Alternativní konektor má jen 9 pinů. Jeden pin se používá k odesílání datových bajtů, druhý k jejich přijímání. Další pin je společnou zemí. Další „užitečné“ piny jsou používány hlavně k signalizaci, přičemž trvale záporné napětí zna-mená „vypnuto“ a kladné „zapnuto“.

Většinu práce má na starosti mikroprocesor UART (Universal Asynchronous Receiver-Transmitter). V moderním provedení je tento mikroprocesor obvykle zabudovaný ještě v jiném mikroprocesoru.

Napětí Jaké napětí má bit

Na sériovém portu EIA-232 je bipolární napětí (vzhledem k zemi kladné a záporné) s hodnotou kolem 12 voltů (někdy jen 5 nebo 10 voltů). +12 voltů na přijímacím i vysílacím pinu znamená bit 0 (anglicky se mu někdy říká „space“) a -12 voltů znamená hodnotu bitu 1 (anglicky někdy „mark“). Tento způsob se někdy nazývá „inverzní logika“, neboť obvykle znamená hodnota bitu 0 nepravdu a zápornou hodnotu, zatímco 1 je jak pravda, tak i kladná hodnota. I když přijímací a vysílací mají opačnou logiku, jiné piny (které řídí modemové linky) mají normální logiku s kladným napětím pro „pravdu“ (resp. pro „zapnuto“ nebo „nastaveno“) a záporným napětím pro „nepravdu“ (resp. „vypnuto“ nebo „není nastaveno“). Nulové napětí nemá žádný význam (snad s tou výjimkou, že může znamenat „vypnutý počítač“).

Je přípustný určitý rozsah napětí. Norma říká, že přenášený signál by měl být mezi 5 až 15 volty a nikdy by neměl překročit 25 voltů. Napětí pod 3 volty není definované (některé terminály ovšem akceptují i nižší napětí jako platné). Občas se setkáte s nepravdivým tvrzením, že běžně používa-né napětí je 5, nebo dokonce jen 3 volty, avšak obvykle je to 11–12 voltů. Používáte-li na Maco-vi port EIA-422 jako EIA-232 (se speciálním kabelem) nebo jako EIA-423, napětí bude skutečně pouze 5 voltů. V dalším popisu ovšem budeme mít za to, že napětí je 12 voltů. V otázce napětí panuje na Internetu dost velký zmatek.

Poznamenejme ještě, že normální logika počítače je pouze několik voltů (standard býval 5 V), takže pokud byste zkusili použít

měřicí přístroj určený pro testování 3–5 V logiky počítače (TTL) na napětí 12 voltů, mohli byste jej poškodit.

Jaká je posloupnost napětí v bajtu

Vysílací pin (TxD) má bez zatížení, když se nic neposílá, -12 voltů (mark). Na začátku bajtu se napětí změní na +12 V (space), což je startovací bit, a zůstane na tomto napětí po dobu trvání jed-noho bitu. Pak jde datový bit nejnižšího řádu. Je-li roven 0, nic se nezmění a na lince zůstane +12 V po dobu dalšího bitového intervalu. Pak přijde další bit atd. Nakonec může přijít paritní bit a pak -12 V (mark) jako stopbit. Na lince zůstává -12 V (nic se nepřenáší) až do příchodu dalšího star-tovacího bitu. Všimněte si, že napětí se nevrací na nulu, a mají-li shodnou polaritu (tj. oba jsou rovny 0 nebo 1), neexistuje jednoduchý způsob (kromě synchronizačního signálu), jak říct, kde končí jeden bit a začíná druhý.

Druhý stopbit bude také -12 V, stejně jako první. Jelikož hranice mezi těmito bity není vyznačena žádným signálem, jediným výsledkem přenosu druhého stopbitu je skutečnost, že na lince bude napětí -12 V dvakrát tak dlouho. Příjemce nemá způsob, jak by zjistil rozdíl mezi druhým stopbi-tem a delším časovým intervalem mezi bajty. Když se na jednom konci používá jeden stopbit a na druhém dva stopbity, komunikace funguje, avšak přenos s jedním stopbitem je rychlejší. Velmi zřídka se také používá 1 x stopbitu, což znamená, že na lince se nechá -12 voltů po dobu 1 inter-valu (tedy stopbit je o 50 % delší).

Co je to parita

Normálně jsou znaky přenášeny jako 7 nebo 8 datových bitů. K nim může (a nemusí) přibýt jeden paritní bit, z čehož plyne délka bajtu 7, 8 nebo 9 bitů. Některé starší terminály a emulátory nepři-pouštějí 9 bitů. Některé nepřipouštějí 9 bitů pouze v případě, když se používají dva stopbity (tím by celková délka vzrostla na 12 bitů, což je moc).

Parita může být lichá, sudá nebo žádná (parita mark a space může být na některých terminálech volitelná). Při liché paritě je paritní bit nastaven na takovou hodnotu, aby počet jedničkových bitů v bajtu (včetně paritního bitu) byl lichý. Změní-li se v důsledku chyby hodnota některého bitu, poškozený bajt bude mít sudou paritu. Obsahuje-li příchozí bit takovou chybu, na obrazovce je vypsán symbol chybného znaku. Sudá parita má analogicky v bajtu sudý počet jedničkových bitů. V průběhu nastavování se počtem bitů na bajt rozumí pouze datové bity (7 pro ASCII a 8 pro různé znakové množiny ISO).

Má-li bit hodnotu 1, nazýváme ji „mark“, hodnotu 0 pak „space“. Je-li hodnota všech paritních bitů rovna 1, hovoříme o „mark paritě“, je-li rovna 0, jde o „space paritu“. Obě tyto parity pouze plýt-vají šíří přenášeného pásma, a je-li to možné, vyhněte se jí. Nemá-li znak paritní bit, hovoříme o znaku (bajtu) bez parity. U terminálů, které neumějí přenášet devítibitové bajty, musíte při osmi-bitové znakové množině vždy zvolit přenos „bez parity“, neboť na paritní bit už není v bajtu místo.

Jak vypadá bajt (rámec)

Při sériovém přenosu bajtů přes porty EIA-232 je nejdříve poslán bit nejnižšího řádu. Sériové porty na PC používají asynchronní komunikaci, při níž je začátek a konec bajtu označen startovacím bitem a stopbitem. Tento způsob nazýváme seskupování bitů a takto seskupené bity, které tvoří bajt, nazýváme někdy rámec. Jeden bajt tedy může obsahovat 9, 10 nebo 11 bitů, přičemž nej-častěji používaný počet je 10. Označení 8-N-1 znamená 8 datových bitů, bez parity, jeden stopbit. Dohromady i se startovacím bitem je to tedy 10 bitů. Téměř zásadně se používá jeden stopbit. Dříve se používaly dva stopbity do rychlostí 110 bitů/sec. (někdy až do 300 bitů/sec.), avšak dnes se používají dva stopbity pouze ve velmi neobvyklých situacích (anebo omylem, příjemce se může domnívat, že dostal dva stopbity).

Nevýhody EIA-232 Pomalé & krátký dosah

Běžně používaný sériový port EIA-232 je svojí podstatou pomalý a značně omezený co do vzdá-lenosti připojení. V reklamě je často uvedeno „vysokorychlostní“, avšak to platí pouze pro velmi krátké vzdálenosti, když je modem umístěn přímo u počítače. Všechny vodiče používají společ-nou zem, takže bez dalšího hardwaru nelze použít kroucenou dvoulinku (nutnou pro vysoké rychlosti). Některé počítače ovšem už mají modernější rozhraní, viz kapitola „Následníci EIA-232“.

Je přímo tragické, že standard RS-232 z roku 1969 nevyšel z technologie kroucené dvoulinky, která je asi stokrát rychlejší. V telefonii se kroucené dvoulinky používají už od 19. století. V roce 1888 (tedy před více než 110 lety) bylo na „Konferenci o kabelech“ přijato doporučení používat pro telefonní systémy technologii kroucených dvoulinek a byly zdůrazněny její přednosti. Po 80 letech pak nebyla tato technologie přijata do standardu RS-232. Vzhledem k tomu, že RS-232 byla původ-ně určena pro připojení terminálů k nízkorychlostním modemům na krátkou vzdálenost, nikdo neuvažoval o vysokorychlostním připojení a o větších vzdálenostech.

Následníci EIA-232

Rozsáhlý popis o jiných portech než EIA-232 naleznete v kapitole Serial-HOWTO v dokumentu „Other Serial Devices“. Pro vysoké rychlosti a velké vzdálenosti byla vytvořena řada standardů na bázi (symetrické) technologie kroucené dvoulinky. Symetrický přenos je až stokrát rychlejší než nesymetrický EIA-232. Při dané rychlosti může být maximální délka kroucené dvoulinky mnoho-krát větší. Zdá se, že některé terminály tuto technologii podporují. I když řada terminálů také pod-poruje porty EIA-423, příliš se neliší od EIA-232 – pouze používají 5 voltů a jsou o něco rychlejší (aniž by používaly kroucenou dvoulinku).

Kroucená dvoulinka zahrnuje EIA-422, EIA-530-A, HSSI (High Speed Serial Interface), USB (Universal Serial Bus) a pochopitelně i ethernet.

Linkové ovladače

Pro textové terminály je rychlost EIA-232 dostatečná, avšak nevyhovující je maximální délka kabelu. Můžete si pomoci symetrickou technologií. Obvyklou metodou, jak přejít na tuto technologii, je nainstalovat linkové ovladače 2@ na sériové linky, abyste převedli nesymetrické na symetrické (a naopak). Jsou to speciální položky, a kupují-li se nové, vyjdou hodně draho.

Synchronní přenos a synchronizace Jak synchronizovat

asynchronní přenos

Vysílací (nebo přijímací) vodič má pouze dva stavy: mark (-12 V) a space (+12 V). Stav 0 neexis-tuje. Posloupnost bitů s hodnotou 1 je tedy přenášena jako trvalé napětí -12 V bez jakýchkoli zna-ček mezi bity. Přijímač musí jednotlivé bity rozpoznávat pomocí časovače synchronizovaného s časovačem vysílače. Časovač „tikne“ při vyslání (nebo přijetí) každého bitu.

Při asynchronním přenosu dosáhneme synchronizace seskupením startovacího bitu, datového bajtu a stopbitu (hardwarově). Příjemce čeká na lince na startovací bit, a když jej rozpozná, spus-tí časovač, který tiká, než přečte dalších 7, 8 nebo 9 bitů. (Ve skutečnosti je to poněkud složitěj-ší, neboť občas se pošlou vzorkové bity, které vyžadují další „tiknutí“ časovače.) Pak se přečte stopbit, hodiny se zastaví a příjemce čeká na další startovací bit. Asynchronní přenos je tak vlast-ně na příjmu synchronizován, avšak mezi bajty synchronizace neexistuje.

Definice asynchronního a synchronního přenosu

Asynchronní (async) znamená „nesynchronní“. V praxi je asynchronní signál to, co posílá a přijí-má asynchronní port: tok bajtů oddělených startovacím bitem a stopbitem. Synchronní (sync) je téměř vše ostatní. Tím ovšem nejsou vysvětleny základní principy.

Teoreticky znamená synchronní to, že bajty jsou posílány konstantní rychlostí, jeden za druhým, v časových intervalech určených tikáním časovače. Pro poslání signálu časovače bývá určen jeden vodič nebo kanál. Asynchronní bajty mohou být odesílány pokaždé jinak, s různými časovými prodlevami mezi jednotlivými bajty (jako když někdo píše znaky na klávesnici).

Existuje hraniční situace, o níž je nutno rozhodnout, zda je synchronním nebo asynchronním pře-nosem. Asynchronní sériový port někdy pošle bajty jako nepřerušovaný tok, což je vlastně syn-chronní přenos, avšak vzhledem k tomu, že jsou přenášeny i start/stopbity (které umožňují nepra-videlné posílání), přenos je asynchronní. Jiným případem je posílání datových bajtů v paketech (bez start/stopbitů) v nepravidelných intervalech mezi pakety. Tento způsob je synchronizovaný, neboť bajty v rámci paketu jsou přenášeny synchronně.

Synchronní komunikace

Přemýšleli jste někdy o tom, k čemu jsou nepoužívané piny na 25pinovém konektoru sériového portu? Většina z nich se používá k synchronní komunikaci, která někdy bývá implementovaná i na osobních počítačích. Některé piny jsou využity pro přenos časových signálů a pro synchronizaci protějšího kanálu. Port EIA-232 může sloužit jak k synchronnímu, tak i k asynchronnímu přeno-su, avšak osobní počítače bývají vybaveny mikroprocesory UART (Universal Asynchronous Rece-iver/Transmitter), např. 16450, 16550A nebo 16650, jež nejsou určeny k synchronnímu přenosu. K tomu je nutno mít mikroprocesor USART (nebo ekvivalent), kde „S“ znamená synchronní. Vzhle-dem k tomu, že trh s vybavením pro synchronní přenos je omezený, synchronní sériové porty budou poměrně dost drahé.

Kromě synchronní části EIA-232 existuje řada jiných synchronních standardů EIA. U tohoto konek-toru jsou 3 piny vyhrazeny časovým signálům. Někdy má za úkol generovat časové signály modem, takže synchronní komunikace bez synchronního modemu není možná (anebo bez zaří-zení, které se nazývá „eliminátor synchronního modemu“, které generuje časové signály).

I když synchronních sériových portů je málo, synchronní komunikace často probíhá po telefon-ních linkách pomocí modemů s korektorem chyb V.42. Ten odstraní start/stopbity a datové bajty uloží do paketů, které jsou pak synchronně přenášeny po telefonní lince.

Blokový režim Úvod do blokového režimu

V Linuxu se blokový režim používá zřídka a jde spíše o historii. Když někdo píše v blokovém reži-mu na terminále, znaky se neposílají na počítač a jsou uloženy do paměti terminálu. Tyto termi-nály také mívají vestavěný editor. Když uživatel stiskne určitou („odesílací“) klávesu, obsah toho, co je uloženo v paměti, se odešle na počítač. Nyní používané editory vi a emacs musí na stisknutí klávesy reagovat ihned, takže blokový režim je nepoužitelný. Dlouhá prodleva mezi stisknutím klávesy a odesláním znaku na počítač, která je průvodním znakem blokového režimu, je pro tyto editory a jiné interaktivní programy nepřijatelná. Blokový režim se tedy už nepoužívá.

Rozhraní starých sálových počítačů IBM používá blokový režim (viz kapitola „Terminály IBM“). Mnohé terminály IBM pracují pouze v synchronním blokovém režimu.

Typy blokových režimů, formuláře

Blokový režim může mít různé podrežimy, např. „stránka“ (jednorázový přenos stránky) nebo „řádek“ (jednorázový přenos řádku). Některé terminály mají jak blokový režim, tak i klasický zna-kový režim a lze mezi nimi přepínat. K asynchronním terminálům s blokovým režimem patří HP2622A, Wyse60, VT130, VT131, VT330, VT340 a Visual500. Řada novějších modelů umí bloko-vý režim emulovat, linuxová konzola to však neumí. Blokové režimy mohou pracovat s formulá-ři. Počítač pošle na terminál formulář, uživatel jej vyplní a stiskne odesílací klávesu (tlačítko). Zpět na počítač jsou poslána pouze data z tohoto formuláře. Formulář samotný (nikoli data) je zobra-zen na obrazovce v chráněných oblastech, které se nepřenášejí na počítač.

Výkonnost

Blokový režim snižuje zátěž centrálního počítače, zejména když je k tomuto režimu hardwarově zkonstruován (jako tomu bylo u sálových počítačů IBM). Ve znakovém režimu je každý zadaný znak ihned poslán na sériový port a obvykle vygeneruje přerušení na centrálním počítači. Ten, jakmile přijme bajt, přeruší všechny ostatní výpočty a přečte tento znak z portu. Dokonce i s UART, který má bufery typu FIFO (zásobník), je hardwarový časový limit obvykle roven době přenosu tří bajtů, takže přerušení je vygenerováno po každém zadaném znaku.

V klasickém blokovém režimu odpovídá jednomu dlouhému přijatému bloku pouze jedno přeru-šení. Pokud je blokový režim použit na klasickém asynchronním sériovém portu FIFO, přerušení je vygenerováno pouze každých 14 bajtů, protože hardwarové bufery jsou šestnáctibajtové. Je tak oproštěn od většiny zátěže a režie nutné ke zpracování přerušení a počítač má tedy v blokovém režimu více času na jiné úlohy.

K významné úspoře v blokovém režimu dochází, když je terminál připojen k počítači po síti. Bez blokového režimu by byl každý zadaný znak (bajt) poslán ve svém vlastním paketu včetně všech režijních bajtů (v paketu TCP/IP používaném na Internetu je to 40 bajtů). V blokovém režimu obsahuje jediný blok velký počet znaků.

Problémy s blokovým režimem

I když blokový režim je výkonnější, je prakticky zapomenut, a to z prostého důvodu. Pro stále rychlejší a levnější počítače není objem přenesených dat mezi počítačem a terminálem kritický. Například modem 56 k generuje stovky přerušení za sekundu (po každých 14 znacích), zatímco když píšete na terminále, dochází jen k několika přerušením za sekundu (při zadání každého znaku). Počet přerušení při psaní tedy není příliš významný (pokud ovšem nemáte k jednomu počítači zapojené stovky terminálů).
Dalším hlediskem je skutečnost, že účinnost při přenosu není podstatná tam, kde uživatel neza-dává z terminálu velké objemy dat. Ukázkovým příkladem, kdy pořizuje množství dat, jsou edito-ry. Avšak pokud editujete v blokovém režimu, musíte používat triviální terminálový editor. Moder-ní editory (vim, emacs) jsou mnohem dokonalejší, avšak nepoužívají blokový režim. Dokonce i v době sálových počítačů s terminály blahé paměti se blokový režim s výjimkou IBM používal poměrně zřídka. Hlavním důvodem byla malá dostupnost softwaru, který by uměl blokový režim využít (opět kromě IBM). Pravděpodobně jej neobsahuje ani databáze terminfo, což značně zne-snadňuje případný vývoj takového softwaru.

Knihy o EIA-232 (RS-232)

(Poznámka: První kniha není zaměřena pouze na EIA-232.)

Black, Uyless D.: Physical Layer Interfaces & Protocols, IEEE Computer Society Press, Los Alamitos, CA, USA, 1996.
Campbell, Joe: The RS-232 Solution, 2. vyd., Sybex, 1982.
Putnam, Byron W.: RS-232 Simplified, Prentice Hall, 1987.
Seyer, Martin D.: RS-232 Made Easy, 2. vyd., Prentice Hall, 1991.

Software pro sériové porty

Odkazy na software pro sériové porty v Linuxu včetně getty a připojení monitorů naleznete na stránkách Serial Software (ftp://ibiblio.unc.edu/pub/Linux/system/serial/).

Poznámky o výrobcích a modelech

Zde naleznete poznámky o výrobcích konkrétních terminálů, jejichž uvádění v jiných kapitolách tohoto návodu by mohlo být újmou na obecnosti výkladu. Máte-li jakékoli informace, které by mohly přispět k popisu určitých terminálů, měly by být zařazeny právě sem. Různé modely mají často společné vlastnosti, jejichž popis by měl být v textu uveden pouze jednou. Bylo by pěkné, kdyby ke každému modelu existovala množina odkazů na podstatnou část dokumentace rele-vantní k tomuto modelu (včetně řídicích sekvencí). Nic takového však neexistuje. Poznamenejme pouze, že některé manuály k terminálům VT (DEC) jsou na Internetu, viz adresa *http://tldp.org/HOWTO/Text-Terminal-HOWTO-23.html#dec_#dec_*. Totéž platí i pro terminály Wyse (*http://tldp.org/HOWTO/Text-Terminal-HOWTO-23.html#wyse_#wyse_*).

Adds

Terminály Adds v menu nekorektně používaly „Xon/Xoff“ pro aktivaci obou způsobů řízení toku
dat. Pro které modely to platí??

Firmu Adds (vyráběla terminály Viewpoint) koupila firma Boundless Technologies.

CIT

Terminály CIT se vyráběly v 80. letech v Japonsku místo terminálů CIE. Koncem 80. let se přesta-ly dovážet. Společnost CIE ještě stále vyrábí (v roce 1997) tiskárny CItoh, avšak nedodává k nim náhradní díly. Některé díly k modelům 224, 326 atd. prodává firma Ernie (Irvine, Kalifornie, USA), tel. (714) 453-9555, nikoli však k modelům 80 a 101.

Nastavovací parametry uložíte v nastavovacím režimu pomocí ^S,. cit80: knoflík pro ovládání kon-trastu je na zadní straně terminálu, cit101e: jas nastavte šipkami v nastavovacím režimu.

Terminály IBM

Nezaměňujte terminály IBM za monitory k IBM PC. Mnohé terminály IBM nepoužívají ASCII, nýbrž osmibitový kód EBCDIC. Pořadí posílání bitů v EBCDIC je údajně opačné, nejdříve jdou bity nej-vyššího řádu. Komunikační standardy sálových počítačů IBM jsou určitý způsob synchronní komu-nikace v blokovém režimu (velké pakety znaků). Za příklad mohou posloužit dva standardy „BISYNC“ a „SNA“ (jež obsahují i síťové standardy). Řada těchto terminálů se připojuje koaxiálním kabelem (RG62A/U) a prostý člověk by se mohl domnívat, že konektor „BNC“ na terminále je určen k připojení na ethernet – není!

I když tento systém IBM je výkonnější než běžně používané systémy v Linuxu, terminály odpoví-dající standardům IBM nejsou použitelné v Linuxu. Nicméně, některé terminály IBM jsou asyn-chronní ASCII terminály a fungují pod operačním systémem Linux na PC. Mohou fungovat napří-klad modely 31xx s výjimkou modelů 317x a 319x, jež nejsou ASCII. Než si pořídíte terminál IBM, přesvědčte se, zda má záznam v databázi termcap (terminfo). Není-li tomu tak, pravděpodobně nebude pod systémem Linux fungovat. Jistotu nemáte, ani když má záznam v této databázi. V termcap například existuje záznam pro 327x, avšak 3270 je synchronní terminál EBCDIC.

Součástí řady 3270 jsou i modely 3278 (konec 70. let), barevný 3279 s grafikou a řadič 3274 (ana-logie 3174). Jsou použitelné s oběma protokoly BISYNC i SNA. Model 3290 má rozdělenou obra-zovku (na čtvrtiny).

Synchronní terminály IBM nejsou připojeny přímo k sálovému počítači nýbrž k „terminálovému řadiči“ (říká se mu také „klastrový řadič“ nebo „komunikační řadič“). Některé řadiče umějí převá-dět synchronní signál na asynchronní, takže synchronní terminál by takto mohl být připojen k séri-ovému portu unixového počítače. Problém s blokovým přenosem však zůstává nevyřešen, viz kapitola „Blokový režim“.

IBM 3153

Port Aux je údajně DCE a je připojen kabelem napřímo.

Dálnopisy

Jde o nejstarší terminály a jsou to historické kusy. Vypadají jako dálkově řízené psací stroje, jsou však větší a hlučnější. Výrobcem byla Teletype Corp., první modely byly vyrobeny ve 20. letech minulého století a předběhly počítače o 30 let. Nejstarší modely používaly mechanická relé a rotač-ní rozdělovače. Pracovaly s pětibitovým kódem Baudot. Viz kniha „Small Computer Systems Hand-book“, Sol Libes, Hayden Books, 1978: str. 138–141 („Teletypes“).

VT (původně DEC, nyní Boundless)

Slavnou řadu terminálů VT včetně nejčastěji emulovaného VT100 vyrobila společnost Digital Equ-ipment Corporation. V roce 1995 prodali obchod s těmito terminály firmě SunRiver, která se nyní jmenuje Boundless Technologies. Podrobné informace o těchto terminálech, manuály a jejich his-torii naleznete na adrese *http://www.vt100.net/*. Další informace jsou na stránkách Shuford's Web-site, *http://www.cs.utk.edu/~shuford/terminal_index.html*. Informace o současné produkci najdete na firemních stránkách Boundless.

VT220: Některé mají BNC konektor pro videovýstup (nikoli pro vstup). Někdy se lidé chybně
domnívají, že je to konektor na připojení ethernetu.
VT510, 520, 525: Plná podpora řízení toku dat DTR/DSR. Některé modely mají „snížené vyzařo
vání“. Model 520 je určen pro několik sezení současně, model 525 umí barevné zvýraznění.

Dorio je model nižší kvality, který umí emulovat mnoho jiných terminálů. Kvalitní je údajně emu-lace konzoly „sco unix“, která odpovídá položce „scoansi“ v databázi terminfo.

Links

Firmu Links převzala společnost Wyse.

Qume

Firmu Qume převzala společnost Wyse počátkem 90. let.

Terminály Wyse

Podrobné informace o starých terminálech na úrovni manuálů viz *http://www.wyse.com/ service/support/kbase/wyseterm.asp.* Naleznete tam specifikace, seznamy řídicích sekvencí, seznamy dílů, často kladené otázky, nastavovací informace atd. Za tyto informace patří firmě Wyse uznání. Ještě starší terminály jsou na stránkách Wyse terminals, *http://www.wyse.com/ products/gpt/ index.htm.*

Firma Wyse byla levnější než ostatní výrobci a získala dominantní podíl na trhu. Ozývaly se sice připomínky ke kvalitě jejich terminálů, zejména se týkaly modelu Wyse 50, avšak velký počet reklamací (i jiných modelů než Wyse 60) je částečně důsledkem značného rozšíření těchto terminálů.

Wyse 50

Nevyráběl se dlouho.

Wyse 60

Nastavení displeje (po odstranění zadního krytu): jas VR202, výška VR302, šířka VR101 (ovlivňuje i výšku). Chcete-li jej používat bez emulace, kódy šipek neodpovídají kódům ve vi (např. ^L). Tento nedostatek odstraníte v režimu „Application key mode“ zadáním řídicí sekvence ESC ~ 3. Šipky pak budou posílat kódy 0xd1 – 0xd4. Vinou chyby v rozhraní příkazu readline v bash shel-lu je nutno opravit soubor /etc/inputrc. Viz kapitola „Chyby v bash shellu“.

Wyse 85

Umí emulovat VT52/VT100/VT200. Chcete-li jej nastavit, stiskněte klávesu F3, šipkami přejděte na požadovanou ikonu a stiskněte mezerník. V zobrazeném menu rolujte pomocí šipek nahoru/dolů. Nastavení můžete kdykoli zopakovat (klávesa F3), aniž byste přišli o původní hodnoty.

Wyse 99-GT

Nastavovací menu terminálu Wyse99GT (konec 80. let). TERM zde znamená „termination“ (znak „ukončení“), nikoli „terminál“.

Jak jsou nastaveny terminály WYSE 99-GT na Kalifornské univerzitě (University of CA), Irvine, USA

David Lawyer, duben 1990

F1 DISP: COLUMNS=80 LINES=24 CELL SIZE=10 X 13 STATUS LINE=STANDARD BACKGROUND=DARK SCROLL SPEED=JUMP SCREEN SAVER=OFF CURSOR=BLINK BLOCK DISPLAY CURSOR=ON ATTRIBUTE=CHAR END OF LINE WRAP=ON AUTO SCROLL=ON

F2 GENERAL: PERSONALITY=VT 100 ENHANCE=ON FONT LOAD=OFF COMM MODE=FULL DUPLEX RCVD CR=CR SEND ACK=ON RESTORE TABS=ON ANSWERBACK MODE=OFF ANSWERBACK CONCEAL=OFF WIDTH CHANGE CLEAR=OFF MONITOR=OFF TEST=OFF

F3 KEYBRD: KEYCLICK=OFF KEYLOCK=CAPS KEY REPEAT=ON RETURN=CR ENTER=CR FUNCT KEY=HOLD XMT LIMIT=NONE FKEY XMT LIMIT=NONE BREAK=170MS LANGUAGE=US MARGIN BELL=OFF PRINTER RCV=OFF

F4 COMM: DATA/PRINTER=AUX/MODEM MDM RCV BAUD RATE=9600 MDM XMT BAUD RATE=9600 MDM DATA/STOP BITS=8/1 MDM RCV HNDSHAKE=NONE MDM XMT HNDSHAKE=NONE MDM PARITY=NONE AUX BAUD RATE=9600 AUX DATA/STOP BITS=8/1 AUX RCV HNDSHAKE=NONE AUX XMT HNDSHAKE=NONE AUX PARITY=NONE (There is a main port (Modem=MDM) and an Auxiliary Port (AUX)

F5 MISC 1: WARNING BELL=ON FKEY LOCK=OFF FEATURE LOCK=ON KEYPAD=NUMERIC DEL=DEL/CAN XFER TERM=EOS CURSOR KEYS=NORMAL MARGIN CTRL=0 DEL FOR LOW Y=ON GIN TERM=CR CHAR MODE=MULTINATIONAL

F6 MISC 2: LOCAL=OFF SEND=ALL PRINT=NATIONAL PORT=EIA DATA SEND AREA=SCREEN PRINT AREA=SCREEN DISCONNECT=60 MSEC SEND TERM=NONE PRINT TERM=NONE PRINT MODE=NORMAL VT100 ID=VT100 POUND=US

F7 tabelátory: Měli byste vidět několik znaků „T“ s mezerami 8 bodů. Pokud neuvidíte, stiskněte „návrat o jeden znak“ (backspace).F8 funkční klávesy: Definice funkčních kláves zde normálně neuvidíte (pokud je někdo

nenasta-vil a neuložil). Znamená to, že budou generovat implicitní znaky (zde nejsou uvedeny).

Ctrl+F5 vypíše „uživatelem definovanou definici“ klávesy F5 atd.F9 zpětná odpověď: Obvykle není definované:

ANSWERBACK =F10 konec: Zadáte-li „DEFAULT ALL“, nastavení od výrobce bude implicitní.NÁVOD k používání

WY-99GT User's Guide: Naleznete jej v WY-99GT Programmer's Guide. Emulace VT100 je známa jako ANSI a používá kódy kláves na str. A-10 a dalších, i kdyby to byl ASCII.Podhlavička na str. A-13 „ASCII Keyboard“ také patří k VT100, protože mohou spadat pod non-ANSI část hlavní hlavičky o několik stránek zpět. Příloha H je v „ANSI Command Guide“ s výjimkou VT52 (ANSI), který je v příloze G.

Wyse 150

Když rušíte nastavení klávesou F12, stisknutím mezerníku v dotazu na uložení nastavení změníte „no“ na „yes“. Věta nalevo od těchto slovíček (no/yes) se týká svislého uspořádání a nemá nic společného s tímto dotazem (menu je zmatené).

Wyse 185

Znakové buňky 10x20. Umí emulovat DEC VT320. Příkon 45 W. Nejnovější modely jsou označeny 185e.

Snížená emise: -ES

ES za číslem modelu znamená sníženou emisi, nižší magnetické pole atd.

Ekologie

Prvním darem je život, druhým láska a třetím porozumění Marge Piercy (http://www.margepiercy.com/)

Ekologie

V této kapitole se seznámíte s postupy, jejichž prostřednictvím lze při používání počítačů se systé-mem Linux co nejméně zatěžovat životní prostředí a šetřit energii či papír. Protože nároky na hard-ware nejsou vysoké, operační systém Linux lze využít i ve starších počítačích a umožňuje tak pro-dloužení jejich životnosti. Některé hry lze využít při ekologické výchově a speciální software umožňuje simulovat ekologické procesy.

Úvod

Přestože počítače lze vnímat jako zdroje znečištění životního prostředí, existují možnosti, jak tyto negativní vlivy minimalizovat. Nedávno jsem se začal zabývat touto problematikou ve vztahu k Linuxu.

Cíle

Některé z cílů tohoto návodu:

Snížení spotřeby energie
Snížení spotřeby materiálu (jako je například papír nebo inkoust)
Snížení produkce odpadu opětovným použitím starších součástí nebo jejich dlouhodobějším používáním
Snížení produkce toxického odpadu, jakým jsou například vybité baterie
Použití Linuxu při ekologické výchově a výzkumu

Sporné body

Některá doporučení uvedená v textu mohou být sporná – například vypnutí zařízení, které se právě nepoužívá. Může se sice zdát, že tím ušetříme energii, ale pokud například časté vypínání a zapínání zařízení snižuje jeho životnost, zatížení prostředí se tím naopak zvyšuje.

Nemám bohužel dostatek odborných znalostí, abych mohl rozhodnout, která z uvedených variant je výhodnější. Některé návrhy mohou různé osoby hodnotit odlišně. Volba je tedy nakonec jen na vás.

Různé

Pokud u jednotlivých programů či balíčků není uvedena adresa URL, můžete ji získat na webu Debi-an (*http://www.debian.org*) nebo ze svého oblíbeného serveru RPM (například *http://rpmfind.net/*).

Snížení spotřeby energie

Systém Linux podporuje různé možnosti, díky kterým lze při práci s počítačem ušetřit energii. Patří mezi ně i tyto: vylepšené řízení spotřeby, určitá nastavení pevného disku, práce bez použití moni-toru a další.

Vylepšené řízení spotřeby (APM/ACPI) Ověření kompatibility systému Linux

Následující text je převzat z dokumentu Battery-Powered-mini-HOWTO (*http://tldp.org/ HOWTO/Battery-Powered/index.html*) a týká se spíše starších počítačů, které podporují APM, u novějších najdete spíše ACPI. „Aby bylo vylepšené řízení spotřeby přenosného počítače funkč-ní, musí systémová paměť BIOS ROM podporovat standardní APM. Kromě toho musí BIOS ROM podporovat jednu z verzí standardu APM 1.0 nebo 1.1 a musí také podporovat 32bitový chráně-ný režim připojení. Upřednostňuje se systém podporující APM 1.1, protože poskytuje více funkcí využívaných ovladači zařízení a podpůrnými nástroji.” Informaci o verzi APM získáte zadáním pří-kazu dmesg a v souboru /proc/apm.

Úvod

Při první instalaci Linuxu bude možná nutné překompilovat jádro. Ve výchozím uspořádání jádra nebude řízení APM pravděpodobně aktivované. Podpora APM se skládá ze dvou částí: podpory jádra a podpory v uživatelském prostředí.

Pro podporu jádra aktivujte parametry v příslušné sekci jádra. U přenosných počítačů nejsou k dispozici všechny funkce. Funkce KONFIG_APM_POWER_OFF by měla být dostupná ve většině přenosných počítačů.

Nástroje podpory APM v uživatelském prostředí naleznete na webu *www.worldvisions.ca/~apen-warr/apmd/.* APMD je skupina programů ovládajících systém vylepšeného řízení spotřeby, kterou nabízí většina novějších přenosných počítačů. Používáte-li jádro verze 2.2.x a chcete experimen-tovat, můžete použít patch jádra, který vytvořil Gabor Kuti (falcon.sch.bme.hu). Díky němu může-te uvést jakýkoli systém Linux do režimu spánku i přesto, že to APM BIOS vašeho počítače přímo nepodporuje.

Upozornění

Pokud máte na stejném disku ještě další operační systém, ujistěte se, že není nainstalovaný nástroj pro „režim spánku“, který by mohl vážně zasahovat do systému Linux – mohl by například používat místo na disku obsazené systémem Linux nebo naopak.

Řešení potíží

Pokud váš počítač pracoval s jádrem verze 2.0.x, ale ne s verzí 2.2.x, poslouží vám rada Klause Frankena (klaus.franken.de): „Pro změnu výchozího nastavení ve verzi 2.2. hledejte příkaz halt v inicializačních skriptech a změňte jej na halt -p nebo poweroff. Podívejte se na man halt – pokud tuto možnost nemáte, potřebujete novější verzi halt.” Naleznete ji v balíčku SysVinit.
Někdy okna (X Window) a APM nespolupracují příliš dobře nebo se počítač zasekává. Steve Rader doporučuje: U některých systémů Linux se servery zasekávají při provádění apm -s. Máte-li tento problém, můžete přejít do virtuální konzoly příkazového řádku a pozastavit chvt 1; apm -s jako root nebo ještě lépe -sudo chvt 1; sudo apm -s. Tyto příkazy zadáte ve skriptu; například my-suspend a poté xapmload --click-command my-suspend.

U některých počítačů (například HP Omnibook 4150 – 366 Mhz) může při přístupu k /proc/apm dojít k chybě jádra „general protection fault: f000“ (obecná chyba ochrany: f000). Stephen Roth-well (*Stephen.Rothwell@canb.auug.org.au*) na stránce *http://www.canb.auug.org.au/~sfr/* vysvět-luje: „Stává se tak proto, že APM BIOS se pokouší použít segment běžného režimu, přestože se nachází v chráněném režimu, což ukazuje na chybu v systému BIOS. S tím jsme se již dříve něko-likrát setkali, pokud se nejednalo o kód vypnutí v režimu BIOS, kde může být řešením návrat do původního režimu, než se pokusíme vypnout počítač. V tomto případě to udělat nemůžeme.”

ACPI

Nejnovějším standardem je ACPI. ACPI4Linux (*http://acpi.sourceforge.net/wiki*) je projekt ovlada-če jádra zaměřený na zavádění úplné podpory ACPI pro systém Linux, včetně ovládání ventiláto-ru, detekce dokování a dokovatelného měřiče teploty WindowMaker.

hdparm (*http://tsx-11.mit.edu/pub/linux/sources/sbin/hdparm-3.0.tar.gz*) je nástroj disku Linux IDE, který vám umožňuje nastavit časové limity otáček a další parametry disku.
Mobile Update Daemon (*http://www.complang.tuwien.ac.at/ulrich/linux/tips.html*). Nahra-zuje standardní démon update. Mobile update démon minimalizuje počet otáček a zkra-cuje dobu používání disku. Uvolňuje vyrovnávací paměť pouze v případě, že disk aktuál-ně provádí i jiné činnosti. Pro zajištění konzistence systému souborů použijte příkaz sync. Jinak by mohlo při výpadku napájení dojít ke ztrátě souborů. Mobile update nevyužívá APM. Proto funguje i ve starších systémech.

Nástroje Toshiba Linux (*http://www.buzzard.me.uk/toshiba/index.html*): Skupina nástrojů systému Linux sloužící k ovládání ventilátoru, hesel správců a funkcí klávesových zkratek notebooků Toshiba Pentium. Součástí je i balíček KDE Klibreta.
LCDproc (*http://lcdproc.omnipotent.net/*). LCDproc je software umožňující přímé zobrazení informací na LCD displeji o velikosti 20x4. Připojuje se pouze k externímu displeji Matrix-Orbital 20x4 LCD, který se připojuje k sériovému portu.
Diald (*http://www.loonie.net/~eschenk/diald.html*). Daemon Diald zajišťuje na základě požadavků připojení k Internetu pomocí protokolů SLIP a PPP. Diald se může dle potře-by automaticky připojit k vzdálenému hostiteli nebo zrušit neaktivní vytáčená připojení.

Jednotka řízení spotřeby (PMU) (Apple PowerBook)

PowerBook nepodporuje specifikaci APM, ale nabízí samostatný protokol pro jednotku PMU (jed-notka řízení spotřeby). Existuje volně dostupný démon s názvem *pmud*, který ovládá řízení spo-třeby; je schopný sledovat stav baterie, uvést počítač do režimu spánku a nastavit různé hladiny spotřeby energie. Vytvořil ho Stephan Leemburg <*stephan@jvc.nl*> a naleznete jej na FTP serve-rech distribuce PPC. Existuje také starší nástroj snooze, který uvádí PowerBook do režimu spán-ku a který je také dostupný na uvedených serverech.

Vypnutí monitoru, užití LED diod na klávesnici

Existuje několik nástrojů, které umožňují získání informací z počítače i bez použití monitoru:
bl – blikající LED diody klávesnice
blinkd – blikající LED diody klávesnice pro telefonní záznamník a fax. Blinkd představu-je pár klient/server, který umožňuje blikání diody klávesnice indikující počet příchozích hovorů v hlasové schránce nebo množství došlých faxů.
mailleds – pomocí diod na klávesnici zobrazuje informaci o nově doručených e–mailech; mailleds vás tichým a nevtíravým způsobem upozorní, že máte nový e–mail – blikáním diody na klávesnici.
tleds – blikání diody na klávesnici indikuje síťové pakety TX a RX. Pokud síťový paket opustí počítač, bliká dioda Scroll Lock. Při přijetí síťového paketu počítačem bliká dioda Num Lock.
ledcontrol (*http://www.iki.fi/sampo.niskanen/ledcontrol/*) – program umožňující zobraze-ní různých informací prostřednictvím běžně nepoužívaných diod na klávesnici. Lze ho kon-figurovat tak, že zobrazí v podstatě jakoukoli přístupovou podmínku true/false nebo indi-kuje libovolné číslo.

Spořiče obrazovky

Je jediným úkolem spořičů chránit obrazovku před vysvícením, nebo také šetří energii? Uveďme si některá doporučení Wade W. Hamptona: Spořiče obrazovky většinou zobrazují grafic-ké prvky, vyhledávají ETI (mimozemský život) nebo provádějí jiné úlohy. Používáte-li spořič obra-zovky tímto způsobem, pravděpodobně spotřebováváte VÍCE energie. Například počítač využíva-jící XSETI jako spořič obrazovky se může zahřát mnohem více (a tudíž spotřebuje více energie), než když slouží k úpravě dokumentu nebo provádí kompilaci. Pokud skutečně chcete energii ušetřit a váš X server a monitor tuto možnost podporuje, použijte v xset možnost dpms (viz ruční nastavení xset). Například pro povolení funkcí DPMS (Energy Star) X serveru: xset + dpms. Režim X serveru můžete rovněž změnit ručně:

xset dpms force standby xset dpms force suspend xset dpms force off

CRT spotřebovává o 25 procent více energie, pokud je obrazovka bílá, než když je obrazovka černá. Proto spořič, který je z větší části černý, šetří energii i bez pomoci DPMS pro vypnutí obrazovky. Energii naopak příliš nešetří velmi světlé či barevné spořiče nebo takové, jež udržují spuštěný procesor.

Příklad programů, které fungují jako spořič obrazovky:

Účelem programu xscreensaver je zobrazovat obrázky v době, kdy se obrazovka nevyží-vá jiným způsobem. Je založen na myšlence, že i pokud s monitorem zrovna nepracujete, měl by dělat něco užitečného. Výhodou tohoto programu oproti kombinaci programů xlock a xautolock je snadná instalace nových grafických prvků: K tomu, abyste mohli při-dat nový režim zobrazení, není potřeba program překompilovat, stačí pouze změnit nasta-vení prostředků. Jako spořič můžete využít jakýkoliv program, který lze spustit takovým způsobem, aby vykresloval hlavní okno obrazovky, aniž by bylo nutné provést jeho úpra-vu. To znamená, že programy, které spustíte jako spořič obrazovky, nemusí být k tomuto účelu přímo určeny.
LockVC je program pro uzamčení konzoly, který spolupracuje se spořičem typu „průlet hvězdokupou". Spuštění LOCKVC ve virtuální konzole spustí hvězdokupu tak, že hvězdy rotují kolem všech tří os.

Nálepka energetické náročnosti (Energy Star)

Robert Horn <*rjh@world.std.com*> napsal: Měl jsem šanci diskutovat tematiku Energy Star s projektanty stolních tiskáren. Shodli se na tom,že dostupná hodnota pohotovostní energie závisí na typu zařízení, ale vyznali se pouze v hod-notách zařízení, jimiž se sami zabývají. Některé z jejich poznámek jsou však zajímavé:

- Hodnocení Energy Star prokazatelně pomáhá šetřit energii. Výjimkou jsou úspory energie založené na časovači. Největších úspor lze dosáhnout používáním zařízení s nízkým únikem proudu (jako je například impulsový motorek). Úspor lze dosáhnout jednotlivými úpravami, ale také rostoucí poptávkou po produktech s nižší spotřebou energie. Díky rostoucí

poptávce po těchto produktech dochází k jejich zlevňování a zkvalitňování. Zastaralý typ přístroje s jedním motorkem (stále běžícím) a možností použít různé nástavce již není nejlevnějším řešením.
Systém hodnocení Energy Star byl dobře vymyšlen. Nikdy nenutil výrobce, aby dělali ústupky týkající se kvality a výkonu výrobků, což téměř znemožnilo protestovat proti změ-nám provedeným za účelem úspor energie během doby, kdy počítač není aktivní. Proto-že k úsporám dochází již od chvíle, kdy se jednotlivé části počítače přestanou pohybovat, jsou tyto úspory jednoznačně prokazatelné.
Hodnocení výkonu počítače je bezpečnostní, ne spotřební. Zdroje na počítači o příkonu 235 W a 300 W tedy určují bezpečnostní limit. Skutečná hodnota plného využití výkonu je mnohem nižší a pohybuje se kolem 20 až 30 procent bezpečnostního limitu. Výrobci také poznamenávají, že je obtížné změřit spotřebu energie vypnutého přístroje. K tomu je zapo-třebí speciálních měřidel proudu. Běžná měřidla střídavého proudu byla vyvinuta spíše pro motory a nejsou vhodná pro spínané zdroje.

Další metody úspory energie

Systém Linux pozastavuje procesor v nečinném cyklu, aby snížil spotřebu energie. První zprávy týkající se systémů OS/2, Win 3.1/95, NET a Linux ukázaly, že Linux spotřebuje mnohem méně energie než operační systémy založené na DOS. Tato skutečnost se však již mohla změnit a její prověření by vyžadovalo nový průzkum.

Většina uživatelů systému Linux by svůj počítač nejraději nechala zapnutý celá dlouhá léta. Něko-lik moderních systémů BIOS však podporuje bezobslužné spouštění a prostřednictvím démona cron můžete dokonce provádět i bezobslužné vypínání. Není tedy nutné nechávat počítač neu-stále spuštěný.

Alternativní zdroje energie – sluneční, větrná a vodní

Přehled odkazů najdete na webu Eklektix (*http://www.cirkits.com*).

Různé úspory

Snížení hluku

Nejhlučnějšími zařízeními počítače jsou ventilátor, pevný disk a reproduktory.

Ventilátor

libsensors0 je knihovna pro čtení teploty/napětí/senzorů ventilátoru.
lm-sensors – ovladače (moduly) jádra pro čtení teploty/napětí/senzorů ventilátoru. Toto je modul pro čtení stavu teploty/napětí/senzorů ventilátoru pomocí čipu LM78/79 a eventu-álně i senzorů na SMBus (Systém řízení sběrnice, většinou v systémech P6 a PII). K dalším podporovaným typům senzorů patří LM80 a LM78 – kopie nazvaná W83781D (viz web *http://www.lm-sensors.nu/*).
ACPI – viz část Vylepšené řízení spotřeby (APM/ACPI).
RTSensors (*http://www.tinet.org/~com.ea/rtsensors/*) může uživatel nastavit jako regulátor. Uživatel může určit maximální a minimální rychlost ventilátorů, maximální povolenou tep-lotu atd. Pokud je teplota v rozmezí nastaveném uživatelem, regulátor sníží rychlost venti-látoru a tím se sníží hluk. Rychlost ventilátoru se tak mění automaticky dle algoritmu, takže už nemusíte používat mechanické či teplotní regulátory.

Pevný disk

Pokud chcete předejít zbytečnému hluku produkovanému pevným diskem, můžete toho dosáh-nout pomocí technik popsaných v části o úsporách energie. Pevný disk je hlavním zdrojem hluku u většiny notebooků. Pevné disky moderních notebooků mají tzv. „řízení zvuku“. Základní před-stavu o možnostech nastavení získáte v příručce. Hlučný pevný disk může být velmi rušivý, pou-žijte man hdparm a zpomalte otáčky disku. Někteří výrobci pevných disků nabízejí nástroje, jako je např. Feature Tool společnosti Hitachi, který umožňuje změnit nastavení Automatic Acoustic Management na nejnižší úroveň hlasitosti (Quiet Seek Mode) nebo na maximální výkon (Normal Seek Mode).

Reproduktory

Pro konzolu použijte setterm blenght 0 a pro X Window xset b off (vypne zvonek).

Úspora materiálu (papíru, inkoustu apod.) Tisk konceptů – více stran na jeden list papíru

K vytištění více než jedné stránky na jediný list papíru použijte nástroje z balíku *psutils*. Tato sada nástrojů slouží k práci s dokumenty PostScript. Podporuje také výběr a uspořádání stran, včetně uspořádání pro tisk brožurek a setřídění pro tisk většího počtu stránek. Většinu těchto funkcí – včetně oboustranného tisku zmiňovaného dále – najdete dnes přímo v tiskovém dialogu grafic-kých aplikací (OpenOffice.org, Mozilla) nebo v nástrojích pro tisk v prostředí KDE (kprinter) nebo GNOME (gtklp), takže použití psutils již není nezbytně nutné.

Stránky HTML nebývají optimalizovány pro tisk. Stránky HTML lze tisknout pomocí nástroje html2ps (konvertor formátu HTML na PostScript). Tento program převádí formát HTML na Post-Script. Kód HTML je možné získat z jedné či více adres URL nebo z místních souborů, zadaných jako parametry v příkazovém řádku. Jazyk HTML je podporován na vysoké úrovni, včetně sekvenčních obrázků, CSS1 a některých funkcí HTML 4.0.

Použít lze také mpage k tisku po 2 či po 4 stránkách (dokumenty PS nebo text ASCII). Můžete tak ušetřit i více než 50 % papíru.

Oboustranný tisk

Jednou z možností úspory papíru je oboustranný tisk. Ben Woodard pracuje na knihovně nazva-né libppd, která vám toto umožňuje již u standardních programů tisku v systému Linux (spolu s dalšími drobnými úpravami – vyladěním tiskárny). Oboustranný tisk je však pro úsporu papíru stále nejdůležitější.
Na internetové adrese *http://sourceforge.net/project/?group_id=1658* si můžete stáhnout beta verzi této knihovny a stejně tak modifikovanou verzi příkazu lpr. Ke stejnému účelu slouží i mpage (*http://www.mesa.nl/pub/mpage*). Z manuálových stránek:

-jfirst[-last][%interval]

Tiskněte jen vybrané listy a očíslujte je (začněte číslem 1). Stránky očíslujte od začátku do konce s intervalem 1. Při zadání -j1-10 se tudíž vybere k tisku prvních 10 listů, -j 1%2 vytiskne jen liché stránky a -j 2%2 vytiskne pouze sudé stránky.

Oboustranného tisku dosáhnete ve dvou krocích. Používáte-li děrovaný papír, vložte ho do tis-kárny tak, aby díry byly v horní části stránky – napravo při vytažení zásobníku s papírem (napří-klad u laserových tiskáren Laser writer II NTX). Liché stránky vytisknete zadáním příkazu

-j 1%2 ...

Zaznamenejte si celkový počet stránek (z daného počtu se vytiskne pouze polovina). Jestliže mpage hlásí po dokončení tisku lichý počet stran, odeberte poslední stránku. Poté vložte papíry do tiskárny tak, aby se text vytiskl na druhou stranu. (Pokud je papír děrovaný, budou díry nyní nalevo). Z naší tiskárny II NTX vychází papír prázdnou stranou nahoru; umístěte ho do zásobní-ku prázdnou stranou nahoru, ale otočený o 180 stupňů. U jiných tiskáren musíte na způsob tisku přijít sami. Nyní vytiskněte stránky se sudými čísly v obráceném pořadí tak, že zadáte:

-r -j 2%2 ...

a budete doufat, že vás nikdo u tiskárny nepředběhl s jinou tiskovou úlohou. Nutno podotknout, že funkce jako tisk sudých/lichých stránek nebo tisk vybraných stránek zvlá-dají novější verze příkazu lpr celkem bez problémů. Více informací najdete v návodu „Tisk v Linu-xu“ v této knize, hledejte parametry page-set a page-range.

Čtení z monitoru místo čtení z papíru

Místo tisku můžete použít jako prohlížeč less/xless/gless. Dokumenty PostScript lze prohlížet s gv a pro dokumenty PDF použijte xpdf nebo acroread (Adobe Reader). Než něco vytisknete, vždy si rozmyslete, zda je to skutečně nezbytné.

Důvody, proč lidé nečtou přímo z monitoru:

Čtení z monitoru je o 30 procent pomalejší (viz studie na webu *http://www.useit.com/ alertbox/9602.html*). Rychlost se zvyšuje s dokonalejším hardwarem (jako je například obrazovka TFT nebo větší obrazovka).
Pro někoho je papírová verze přehlednější. I v tomto případě může být řešením lepší soft-ware (jako je Linux) a hardware.

Někteří lidé si dokumenty přečtou raději prostřednictvím příručního počítače (například PalmIII, Newton Message Pad, Psion 5).

Další metody

Jiným způsobem úspory papíru je užití poznámek a zvýraznění/přeškrtnutí (v případě, že si doku-ment předáváte se spolupracovníkem). Koncept napíšete v programu WordPerfect a pošlete e-mai-lem spolupracovníkovi. Ten vám ho pak pošle aktualizovaný/opravený zpět. Změny lze zvýraznit pomocí některého z nástrojů programu WordPerfect, jako je například zvýraznění/přeškrtnutí. Nové verze textových procesorů podporují týmovou práci např. pomocí sledování změn v doku-mentu (OpenOffice.org).
Pokud je to možné, pořiďte si menší počítače a monitory. Tím ušetříte obalový materiál a vypro-dukujete méně odpadu. Například krabice od monitoru CRT s velikostí 15 palců je 2 až 3krát větší než krabice od stejně velkého monitoru LCD. Operační systém Linux pracuje dobře i s 15palco-vými monitory LCD a menšími počítači, jako například Netwinder, E3000, nebo s barebone systé-my různých výrobců.

Mohli byste však namítnout, že nepohodlné čtení textu z menšího monitoru může lidi vést k tisku dokumentů. Jiní zase tvrdí, že používání monitorů LCD škodí životnímu prostředí více než použí-vání monitorů CRT, neboť se při jejich výrobě používá více toxických látek. Zná snad někdo stu-die a výzkumy řešící tento problém?

Recyklujte papír, inkoust a obalové materiály. Existují kazety do tiskáren, které se dají znovu napl-nit. V Německu jsou označeny nálepkou Blauer Engel. Kazety laserových tiskáren jsou většinou použitelné i poté, co se na panelu zobrazí zpráva, abyste toner vyměnili. Stačí je jen pořádně pro-třepat.

Dokumenty LaTeX: Použitím direktivy \usepackage{ccfonts} nahradíte běžný typ písma písmem se širšími linkami a výraznějšími patkami, což i při nižším rozlišení zlepší čitelnost. Písmo je tmavší (tzn. použije více inkoustu) než písmo typu CM a není tak pěkné. Pro tisk písma o běžné velikosti ho proto nedoporučujeme.
Změna velikosti: Místo psnup nebo pstools doporučuji psnup vytvořený v Perl4 Malco-mem Herbertem (pochází sice již z roku 1994 a delší dobu nebyl nijak upravován, ale exi-stuje jeho nástupce yup, s kterým se můžete seznámit na webu *http:// redback.spyda.net/~mjch/yup/*).

Nabízí mnoho možností, které dovolují nastavit všechny čtyři okraje a mezery mezi sloup-ci zvlášť. Toho lze využít ke zmenšení okrajů, čímž zůstane více místa pro text. Pravděpo-dobně bude nutné trochu experimentovat (zkoušet znovu a znovu nové hodnoty a ověřo-vání výsledků pomocí gv).

Běžně používanými možnostmi jsou:

- -p2 (nebo -p4 atd., jako -2 ve starém psnup),
- -NIH (bez okrasných prvků),
- -l10 -r20 -b30 -t40 (přidat k okrajům),
- -g50 (přidat k mezeře mezi sloupci).

(Tyto hodnoty se mění v závislosti na velikosti papíru a původních okrajů, záporné hod

noty jsou povoleny.)

PDFjam (*http://www.warwick.ac.uk/go/pdfjam*) je malá sada skriptů, která nabízí několik funkcí balíčku pdfpages pro pdfLaTeX. V současné době jsou dostupné tyto nástroje: pdf-nup, pdfjoin a pdf90. PDFjam závisí na běžících instalacích (pdf)LaTeX. pdfnup vkládá více stránek dokumentu na jednu stránku, pdfjoin slučuje více dokumentů PDF a pdf90 otáčí stránky dokumentů PDF. Pro Mac OS X jsou k dispozici ukázky aplikací, které umož-ňují použití myši pro přetahování do skriptu.
Pdftk je nástroj podobný PDFJam. Umožňuje pokročilou manipulaci se soubory PDF. Další zajímavou variantou je *PDFedit* (*http://pdfedit.petricek.net/*) z českých luhů a hájů.
Různé typy inkoustových tiskáren mohou tisknout na zadní stranu již potištěného papíru. Vyzkoušejte různé výrobce. Starší inkoustové tiskárny Canon poskytují 360 dpi, starší tis-kárny HP pak 300 dpi. Pokud za použití LaTeX s velikostí 10 bodů vytisknete 4 listy na stránku, čitelnost bude někde mezi těmito dvěma hodnotami.
*Další operační systémy: P*okud musíte pracovat s MS Windows, měli byste si pořídit origi-nální ovladač Adobe PostScript Driver, než abyste používali ten v MS Windows. Tyto ovla-dače umožňují vložit více listů na jednu stranu. V současné době oba programy psnup nefungují v Adobe PS, MS Windows PS a PostScript získaném v souborech PDF systému MS Windows.
Moderní počítačová písma (bez německé diakritiky) jsou k dispozici jako písma TTF na serverech CTAN. Použití těchto písem vám umožní zvýšit estetickou hodno-tu dokumentů a také ušetřit místo na papíru.
Ghostscript nabízí nový formát výstupu pswrite, který vytváří výstup ve správném formá-tu PostScript. Tuto funkci můžete použít k opravě poškozeného formátu PostScript (napří-klad z ovladačů společnosti Microsoft) tak, že je povoleno jejich přepracování pomocí psnup.
impose+ (*http://imagic.weizmann.ac.il/~dov/freesw/impose+/*) je sada nástrojů pro manipu-laci s PostScriptem. Hlavním programem je *impose* pro tisk formátu PostScript, který je kompatibilní s DSC. Pokouší se odstranit bílý prostor z výtisku tím, že zkouší propojit původní PostScript s okraji tištěné oblasti. Výstup pak vypadá lépe než zjednodušené roz-ložení původních neupravených stránek.
hpijs (*http://www.hpgs.cjb.net/*) je ovladač tiskárny umožňující tisk na tiskárnách řady 6xx společnosti HP s využitím ekonomického režimu. Ovladače obsažené v GhostScript zajiš-ťují vše kromě práce v ekonomickém režimu.
Duplex (*http://sourceforge.net/projects/duplexpr/*) je sada skriptů sh umožňující oboustran-ný tisk u tiskáren, jejichž hardware tuto funkci nepodporuje (tiskárny s archovým zásob-níkem). Je určen pro tiskárny připojené k pracovním stanicím. Může být použit přes rouru (pipe) a aplikace jej může používat jako ovladač oboustranného tisku. Jeho jedinečnou vlastností je schopnost tisknout mnoho oboustranných úloh najednou, přičemž se nejprve vytisknou stránky s lichými čísly všech úloh a pak stránky sudé.

Ekologické chování se vyplácí

Programy psutils nejenže šetří papír, ale jsou skvělým nástrojem k vytváření vyhovujícího vzhle-du stránky. Představte si pěknou vázanou příručku formátu A5 namísto ledabyle sešitého svazku listů formátu A4!

V závislosti na délce slov a délce odstavců může i místo na papíru ušetřit i rozložení textu do několika sloupců (v textu se pravděpodobně objeví více rozdělených slov, na druhou stranu se sníží počet neúplných řádků). Tímto způsobem příliš papíru neušetříte, možná vměstnáte 2,1 stránky na 2,0. Pomocí psnup stejné množství textu umístíte na jednu stránku. Text rozložený do více sloupců se také lépe čte.

psdim (*http://www.mathstat.dal.ca/~selinger/psdim/*) je nástroj, který se používá společně s nástrojem pstops. Hledá v obsahu dokumentu PostScript délku tištěných stran. Z toho pak vypo-čítává optimální umístění stránek pro tisk.

Recyklace materiálu (papíru, kazet do tiskáren, CD, disket, pásek)

Všechen tento materiál je recyklovatelný. Hledejte ve svém okolí společnosti, které se recyklací zabývají. Sami můžete začít třídit jednotlivé druhy odpadu.

Snížení radiačního, elektromagnetického a tepelného záření

Staré CRT monitory jsou zdrojem radiačního a elektromagnetického záření. To lze snížit pomocí olověných filtrů, obrazovek LCD. Některá označení ekologicky šetrných výrobků (například TCO95) v sobě zahrnují i nejvyšší přípustnou hodnotu záření.
Zejména pokud je v místnosti více počítačů, mohou vyzařovat tolik tepla, že se místnost ohřívá natolik, že musí být chlazena/klimatizována. Tepelné ohřívání lze omezit vypínáním počítačů.

Prodloužení životnosti hardwaru

Recyklace hardwaru

Trh s počítači ovládají do značné míry dodavatelé, kteří se snažící se prodat nový software a hard
ware. Neexistuje žádný komerční marketingový nástroj podporující jejich opětovné používání. Operační systém Linux nemá žádné speciální nároky na hardware, postačí vám tedy i starší hardware.

Podporované typy procesorů

Linux je kompatibilní s procesory řady Intel, včetně 386, 486, Pentium, Pentium Pro a Pentium II a také s procesory AMD, Cyrix a dalšími. Linux zatím nepodporuje procesory řady 286. Řešením se zabývá projekt ELKS (*http://elks.sourceforge.net/*). Pokud chcete, můžete použít Minix – jeden z předchůdců systému Linux. Minix podporuje procesory 8088 až 286 s velikostí paměti od pou-hých 640 kB. Podrobnější informace o systémech podporovaných jádrem Linux získáte na webu *http://tldp.org/FAQ/Linux-FAQ/index.html.* Rychlou a nízkoenergetickou alternativou je procesor ARM. Na procesoru ARM je založen například Corel/Rebel Netwinder (viz *http://www.rebel.com/* a *http://developer.intel.com/design/strong/*).

Aplikace a distribuce pro starší počítače Projekt RULE

Hardware je pouze tak starý, jak starý je používaný software. Projekt RULE (*http://www.rule -project.org/*) se snaží vytvořit moderní, volně přístupný software použitelný na počítačích starých pět i více let, na kterých by nebylo možné nainstalovat současné distribuce Linuxu nebo by běže-ly příliš pomalu.

ISDN Router

Projekt *ISDN Router* (*http://schumann.cx/isdn-router/*) vám umožní změnit starý hardware v bez-pečný ISDN směrovač s překladem adres, včetně ukládání do mezipaměti jmenného serveru, služ-by IP Port Forwarding a spojování více ISDN modemů. Systém se vejde na disketu a uživatelé mohou měnit konfiguraci pomocí jednoduše ovladatelné nabídky (v konzole nebo prostřednic-tvím protokolu telnet).

Linux LiveCD

Linux LiveCD vám umožňuje sdílet a zabezpečit bránou firewall vysokorychlostní připojení k Inter-netu a používat technologii WiFi. Funguje u DSL, modemu, T1 i vytáčeného připojení a podpo-ruje levný hardware, jako je například USB/PCMCIA WiFi a síťové karty. Požadavky na hardware: počítač s minimálně touto konfigurací – procesor 486, 16 MB RAM, dvourychlostní jednotka CD--ROM, disketová jednotka, 1 nebo 2 síťové karty, BEZ pevného disku(!). Volitelně můžete použít i kartu WiFi.

DeLi Linux

DeLi Linux (*http://www.delilinux.de/*) je součástí distribučního balíčku Linux, který je určen pro starší počítače s procesory řady 486 až po Pentium MMX. Je zaměřen na používání ve stolních počítačích. Obsahuje e-mailové klienty, grafický internetový prohlížeč a kancelářský balíček s tex-tovým a tabulkovým editorem. Pro úplnou instalaci, včetně X.org a nástrojů pro vývoj, postačí 300 MB volného místa na pevném disku.

Gentoo na starých počítačích

Na starých počítačích nemusíte používat starý software. Starý software obsahuje chyby a není dobře zabezpečený. Většina nových distribučních balíčků bude po spuštění na těchto počítačích velmi pomalá; důvodem není nový software, ale to, že očekávají, že je spouštíte na novém počí-tači, a automaticky nainstalují velké množství softwaru. Snahou projektu Gentoo-on-old-hardware (*http://gentoo-wiki.com/HARDWARE_old*) je dosáhnout odlehčeného, minimalistického systému, který spustí jen to, co potřebujete, a nic víc; na rozdíl od mnoha operačních systémů a obsáhlej-ších distribucí, ke kterým patří například Fedora.

FreeS/WAN

Linux FreeS/WAN (*http://www.freeswan.org/*) nabízí rozšíření jádra IPSEC (IP Security, což je šif-rování i ověřování) a IKE (Internet Key Exchange; démon pro vytváření klíčů a šifrované směro-vání), ale i různé skripty rc a dokumentaci. To umožňuje správci systému Linux budovat brány VPN i ze starých 584 a 486. Verze 1.00 spolupracuje s dalšími systémy IPSEC a IKE

vyvinutými jinými dodavateli, jako je například OpenBSD.

Tiskový server

Starší počítače obvykle slouží jako tiskový server.

Práce s omezenými prostředky a ladění systému

Související dokumenty

LBX-HOWTO (*http://tldp.org/HOWTO/LBX.html*) LBX (Low Bandwidth X) je rozšíření X serveru, který provádí kompresi protokolu X Window. Má se používat spolu s X aplikace-mi a X serverem, které jsou rozděleny pomalým síťovým připojením za účelem zdokona-lení zobrazení a doby odezvy.
Small-Memory-HOWTO (*http://tldp.org/HOWTO/Small-Memory/index.html*) popisuje, jak spustit Linux na systému s malou pamětí.

Nedostatek místa na disku

Existují různé způsoby, jak získat více místa na disku (například sdílením místa, uvolněním nepoužívaného a nepotřebného místa, vyladěním systému souborů a komprimace). Některé z těchto metod vyžívají paměť a ne místo na disku. Jak uvidíte, je pro uvolnění místa nezbytné provést mnoho kroků.

Metody

- Stripping: Ačkoli mnoho distribucí obsahuje „stripované“ binární soubory, je dobré tuto skutečnost zkontrolovat. Podrobnosti naleznete v man strip. K vyhledání můžete použít příkaz file nebo nástroj findstrip. Pozor: Nestrippujte knihovny, někdy se kvůli chybné technice programování odstraní nesprávné symboly.

  Russell Marks <*rus@beeb.net*> doporučuje:

  V dnešní době mnoho lidí kompiluje za použití parametru -g, což mi připadá nevhodné
  (ve skutečnosti tento postup ubírá místo).

  strip nabízí možnost --strip-debug, která odstraní pouze ladicí symboly. To je v mnoha
  případech dostačující a dobré pro použití v knihovnách. Nedávno jsme pracovali s nástrojem SuSE 6.3, takže si můžete prohlédnout příklad:

```
bash-2.03# cd /lib bash-2.03# ls -l libc.so.6 -rwxr-xr-x 1 root root 4223971 Nov 6 16:22 libc.so.6 bash-2.03# strip --strip-debug libc.so.6
bash-2.03# ls -l libc.so.6 -rwxr-xr-x 1 root root 1200355 Dec 8 00:13 libc.so.6
```

Perforace: zum(1) přečte seznam souborů ze stdin a pokusí se tyto soubory perforovat. Perforace znamená, že řady nulových bajtů nahradí lseek, a díky tomu nemá systém sou-borů šanci přidělit těmto bajtům skutečný prostor na disku. Příklad: find . -type f | xargs zum. Detaily o dírách v souborech najdete v části „Příručka administrátora“, kapitola „Sou-borové systémy“.
Odstranění nepotřebných a duplicitních souborů: Vyhledejte v systému soubory jádra, záložní soubory editorů emacs (<#SOUBOR#>) a vi (<SOUBOR>.swp), dále záložní soubory rpm (<SOUBOR>.rpmorig) a patch. Při hledání duplicitních souborů můžete zkusit příkaz finddup. Zvolte si určitý systém pro názvy záložních, dočasných a testovacích souborů (jako je například značka za názvem souboru).
Odstranění dočasných souborů: Například /tmp, existuje k tomu i nástroj tmpwatch.
Zkrácení logů: obvykle soubory v adresáři /var/log. K tomuto účelu je k dispozici něko-lik šikovných nástrojů, jako je například savelog.
Odstranění souborů: Odstraňte soubory, které nejsou „nezbytné“, jako je dokumentace /usr/doc a zdroje, například /usr/src.
Nepotřebné knihovny: Pro vyhledání nepoužívaných knihoven můžete použít nástroj bin-stats.
Systém souborů: Zvolte systém souborů, který ekonomicky nakládá s prostorem na disku, například rsfs neboli Reiser Filesystem.
Vylaďte svůj systém souborů například prostřed-nictvím nástroje tune2fs. Vyberte přiměřenou velikost oddílu a bloku.
Zmenšení jádra: buď použitím jen potřebných funkcí jádra a/nebo vytvořením komprimo-vaného obrazu jádra (použijte make bzImage).
Komprimace: Systém souborů můžete komprimovat prostřednictvím nástroje gzip a dekomprimovat jej během práce. Případně můžete komprimovat pouze určité soubory. Komprimované soubory můžete spouštět pomocí nástroje zexec.

- Komprimované souborové systémy:

Pro systémy souborů e2fs je k dispozici verze komprimace e2compr (viz *http://e2compr.sourceforge.net/*).
DMSDOS (*http://cmp.felk.cvut.cz/~pisa/dmsdos/*), který umožňuje vašemu počítači pří-stup ke komprimovaným jednotkám Windows95 (DriveSpace, DoubleStacker). Nepo-třebujete-li kompatibilitu DOS/Windows95, tj. chcete komprimovat pouze data Linux, není tento postup tvůrcem programu doporučován.
Jiné komprimované souborové systémy jsou například *SquashFS* nebo *JFFS2* (detaily viz článek *http://lwn.net/Articles/219827/*).
Sdílení oddílů: Můžete sdílet odkládací prostor či datové oddíly mezi různými operačními systémy (viz mount). Pro připojení komprimovaných jednotek MS-DOS Windows95 (DoubleSpace, DriveSpace) můžete použít DMSDOS.
Knihovny: Použijte jinou (starší) knihovnu (například libc5), která bývá menší než kni-hovna libc6 neboli glibc2.
Jádro: Pokud vašim potřebám vyhovuje starší verze jádra, můžete ušetřit určitý prostor na disku.
Grafické uživatelské rozhraní: Pokud je to možné, používejte co nejméně grafického uživatelského rozhraní.
Malé distribuce: K dispozici je několik distribučních balíčků, které se vejdou na jednu dis-ketu 3,5” nebo zabírají maximálně 10

MB prostoru na disku. Více informací naleznete na internetové adrese *http://tuxmobil.org/howtos.html.*
Tailmerging for Ext2 (*http://www.innominate.org/~phillips/tailmerge/*): Slučování konco-vých bloků souborů je metoda, která pomáhá šetřit prostor v systému souborů s rozsáhlý-mi bloky a mnoha malými soubory. Tailmerging for Ext2 je nadstavba nástroje ext2, který slučuje koncové bloky několika souborů do jednoho sdíleného bloku. Detaily k tématu najdete v části „Příručka administrátora" v kapitole „Velikost bloku v souborovém systému".

Rychlost pevného disku

Výkon pevného disku můžete zvýšit pomocí nástroje hdparm. Při zkoušení programu pozor – podívejte se *důkladně* do manuálových stránek!

Malá pamě Metody

Velikost paměti zjistíte pomocí nástrojů free a top nebo zkuste projekt *Mergemem* (*http://www.complang.tuwien.ac.at/ulrich/mergemem/*). Mnoho programů obsahuje *oblasti paměti se stejným obsahem*, které operační systém nezjistí. Obvykle obsahují data, jež se vygenerovala při spuštění a dlouhou dobu se nezměnila. Nástroj Mergemem tyto oblasti vyhledává a umožňuje jejich sdílení. Sdílení se odehrává na úrovni operačního systému, přičemž je skryté úrovni uživatelských programů. Nástroj Mergemem je užitečný zejména v případě, že spouštíte více instancí překladačů a emulátorů (jako je například Java nebo Prolog), protože dokáže udržet kód v oblasti důvěrných dat. Využít toho mohou i další programy, ačkoli na nižší úrovni. Můžete také zmenšit velikost jádra odstraněním všech nástrojů, které pro vás nejsou nezbytné a také co největší modulací jádra. Mož-ností je také vypnutí všech nepotřebných služeb a démonů (například lpd, mountd, nfsd) a zavře-ní některých virtuálních konzol. Samozřejmě můžete kdykoliv použít odkládání. Pokud je to možné, použijte zdroje jiného počítače, například pomocí X, VNC, nebo dokonce telnetu. Více informací najdete na webu Virtual Network Computing (*http://www.realvnc.com/*).

Nízká rychlost procesoru

Pokud budete chtít zvýšit rychlost procesoru přetaktováním, mohli byste tím však poškodit hardwa-re. Některé příklady lze najít na webu Adorable Toshiba Libretto – Overclocking (*http://www. silverace.com/libretto/*).

Menší aplikace a produkty

BOA – Vysoce výkonný webový server. To znamená, že na rozdíl od klasických serverů nevětví každé příchozí připojení. Vnitřně násobí všechny odchozí připojení HTTP a větví pouze programy CGI (což musí probíhat postupně). Předběžné testy ukazují, že boa zvlád-ne s procesorem Pentium 100 MHz několik stovek požadavků za sekundu.
MGR – grafický systém, který používá mnohem méně prostředků než X Window.
Low Bandwidth X – v únoru 1998 napsal Alan Cox v LINUX REDUX: „... existují 2 způso-by výborné práce s *normálními* aplikacemi. LBX (Low Bandwidth X) je *oficiální* aplikace konsorcia X (nyní OpenGroup na webu *www.opengroup.org*). Dxpc (*http://ccwf.cc. utexas.edu/~zvonler/dxpc*) je alternativou, kterou upřednostňuje většina uživatelů. Tyto

> systémy se chovají jako proxy servery pro X Window a komprimují tok dat až na 50 pro-cent u běžných požadavků, ale lze dosáhnout i snížení na 25 procent původní šířky pásma. S dxpc jsou aplikace celkem použitelné prostřednictvím připojení s modemem 28,8 kHz nebo v síti Internet."

Blackbox – Jedná se o správce oken pro X. V mnoha aspektech se podobá oblíbeným pro-středím, jakým je například Windows Maker, Enlightenment a FVWM2. Tento správce by pro vás mohl být zajímavým, pokud vás již unavují správci oken, kteří příliš zatěžují vaše systémové zdroje, přesto však požadujete atraktivní a moderní rozhraní.
Linux-lite – produkt založený na typu jádra verze 1.x.x. Vhodný pro systémy s velikostí paměti pevného disku pouze 10 MB (viz výše uvedená adresa).
SmallLinux – (*http://www.superant.com/smalllinux/*) – odlehčená verze systému Linux a nástrojů na třech disketách. Typ jádra 1.2.11. Základním formátem disku je ext2 a k dis-pozici jsou nástroje fdisk a mkfs.ext2, takže lze provést instalaci na pevný disk. Je vhod-ný pro starší počítače s operační pamětí RAM menší než 4 MB.
cLIeNUX – produkt orientovaný na klientské použití systému Linux.
minix – toto není Linux, ale minimalistický UNIX. Je vhodný pro velmi malé systémy, napří-klad s typem procesoru 286 a operační pamětí RAM 640 kB. Více informací naleznete na internetové adrese *http://www.cs.vu.nl/~ast/minix.html.* Existuje pro něj i implementace X Window s názvem mini-x (autorem je David I. Bell a podrobné informace jsou k dispo-zici na webu ftp://ftp.linux.org.uk/pub/linux/alan/).
screen – drobný, ale účinný správce konzoly. John M. Fisk <*fiskjm@ctrvax.vanderbilt.edu*> napsal na webu LINUX GAZETTE (*http://www.linuxgazette.com/issue01to08/ lg_issue7.html#screen*): „Celé je to jen o grafickém uživatelském rozhraní" – tedy něco v tom smyslu, že hlavní výrobci operačních systémů by si měli získat vaši důvěru. Pravdou je, že v některých případech příkazový řádek zůstává velmi dobrou volbou k provedení určité úlohy. Je rychlý a vcelku pohotový a jeho používání je vhodné zejména u počítačů s omezenou velikostí paměti či výkonností procesoru. Nezapomínejte, že mnoho úkonů lze provádět v konzole. screen je správce okna celé obrazovky, který rozděluje fyzický ter-minál pro různé procesy, přičemž se obvykle jedná o interaktivní shelly. Každý virtuální terminál nabízí funkce terminálu DEC VT100 a k tomu několik funkcí ze standardů ANSI X3.64 (ISO 6429) a ISO 2022 (například vložení/smazání řádku a podpora různých znako-vých sad). Podpora více uživatelů,

rozdělení obrazovky, podpora emulace stavu hardwa-ru, konfigurovatelný oddělovač oken, permanentní oddělovač oken, časová razítka soubo-ru protokolu, volitelný vestavěný protokol telnet, nastavitelná podpora slepeckého písma, podpora komprimace.
tinyirc – klient IRC. Nenabízí většinu z pokročilejších příkazů klientů typu ircII, ale fungu-je.
tinyproxy – jednoduchý server proxy vyvinutý k práci s využitím minima systémových pro-středků. Díky této jednoduchosti je tinyproxy ideální pro úpravy uživatelského prostředí – zdroj tohoto nástroje je velmi srozumitelný, a tudíž můžete pomocí jednoduchých příkazů začít přidávat funkce podle vlastních potřeb.

Další metody

Baterie Ni-Cad je nutné pravidelně vybíjet, aby se prodloužila jejich životnost. Baterie, jako jsou Ni-Cad, Lead Acid a NiMH, obsahují TOXICKÉ chemikálie. Mělo by se s nimi zacházet tak, aby jejich životnost byla co nejdelší, a poté by se měly recyklovat a ne jednoduše vyhodit do koše.
Jednou z technik, jež stojí za zmínku, je „železná“ baterie zmíněná v článku na webu *http://news.excite.com/news/990815/01/science-battery-iron.* Tento typ baterie by mohl řešit pro-blém vzniku toxických odpadů z použitých baterií, ačkoliv se takové baterie pravděpodobně dočkáme až za několik let.

Patch BadRAM v systému Linux

Cílem opravného balíčku BadRAM Patch (*http://rick.vanrein.org/linux/badram/*) je spustit jádro systému Linux tak, aby mohlo obejít poškozené části paměti RAM. Poškozená paměť RAM může obsahovat chybné bity v některých (známých) adresách. Obvykle se taková paměť RAM považu-je za nepoužitelnou a vyhodí se; čím větší paměť RAM, tím vyšší šance selhání adres. S neustále se zvětšující pamětí RAM by proto bylo příjemné mít jinou alternativu, než je vyhození poškoze-ných čipů paměti RAM.

Instalace systému Linux na starší hardware

Lightweight Linux, první část (*http://www.128.ibm.com/developerworks/linux/library/l-lwl1/*): Hardware je pouze tak starý, jak starý je používaný software. Moderní operační systém a aktuální aplikace vrací staršímu hardwaru výkonnost. V této části vám představíme nejvhodnější postup, jehož prostřednictvím lze nainstalovat funkční systém Linux na starším hardwaru nebo na moder-ním hardwaru s omezenou velikostí paměti a úložiště dat.

Vyprázdnění nepotřebných lokalizačních souborů: localepurge je jednoduchý skript pro Debian sloužící k uvolnění místa na disku, které zbytečně zabírají nepotřebné lokalizační soubory a loka-lizované manuálové stránky. V závislosti na instalaci můžete ušetřit 20, 30 nebo i více megabajtů, obvykle určených soubory tohoto typu.

Inovace a oprava hardwaru

Výrobci obyčejně negarantují záruku pokud kryt otevřel někdy jiný než autorizovaný personál. Pokud to i přesto chcete zkusit, nabízíme vám několik zajímavých odkazů na weby, které se zabý-vají opravami, rozebráním, inovací nebo úpravou notebooků, opravami rozbitých PDA a příruč-ních počítačů (Hand-Held) (*http://repair4laptop.org/, http://repair4pda.org/*). Můžete také opravo-vat mobilní telefony, kapesní přehrávače hudby a videa, opravit a upravit myš či opravit tiskárnu a zásobníky pro inkoust nebo toner (*http:repair4mobilephone.org/, http://repair4player.org/, http://repair4mouse.org/* a *http://repair4printer.org/*).

Jiné operační systémy

Se svolením George Whita <*gwhite@bodnext.bio.dfo.ca*>: Můžete si koupit použitý počítač (SGI, Sun, NeXT) s operačním systémem Unix, který je schopen pracovat s širokou škálou open-sour-ce softwaru. V některých případech (SGI Indigo2) můžete i nadále používat stávající verzi ope-račního systému, jindy bude vhodnější volně dostupný operačního systém, jakým je například Linux, vždy však máte přístup k velkému množství dobrého softwaru a k nástrojům, abyste mohli vytvořit vlastní. Nižší spotřeba energie u starších počítačů znamená delší běh na UPS, případně můžete použít malé náhradní zdroje energie.

X10 – domácí automatizační systém

Moduly X10 jsou zařízení, která se dají zapojit do elektrické zásuvky a umožňují například na dálku zapnout připojenou lampu nebo spotřebič. Existují také moduly X10, které se umístí jako vypínače na zdi a zapínají či vypínají světla, jiné mohou sloužit k spínání termostatu.
Zařízení příbuzné modulům X10 (*http://www.x10.com/*) se nazývá Firecracker a je skvělým ochranným zařízením. Umožňuje ovládání zařízení X10 prostřednictvím sériového portu. Progra-my, jako je bottlerocket a gtk-x10, umožňují programům systému Linux ovládat zařízení X10 pomocí Firecracker.

GNU Phantom.Home (*http://www.joethielen.com/phantom/home/*) je počítačem ovládaný domácí automatizační systém. Software se skládá ze schématu obvodu pro vytvoření ovladače Phan-tom.Home, což je jednoduchá obvodová deska připojená k paralelnímu portu počítače. Pomocí kombinace hardwaru a softwaru lze ovládat (například přepínat) až 120voltové zařízení. Úpravou obvodové desky podle vlastní potřeby a s určitými znalostmi elektroniky můžete ovládat praktic-ky každé zařízení s

jakýmkoli napětím.

Nepřerušitelný zdroj napájení (UPS)

Pokud žijete či pracujete v oblasti s častým výskytem bouřek, měli byste mít k dispozici nepřeru-šitelný zdroj napájení (UPS). Předejdete tak poškození monitoru, procesoru nebo modemu pře-pětím při úderu blesku. Chráníte tak hardware, software a šetříte svůj čas i peníze. Podrobnosti naleznete v dokumentu UPS-HOWTO (*http://tldp.org/HOWTO/UPS-HOWTO.html*).

UPS chrání váš hardware, šetří práci atd. Chrání hardware i v oblastech s častým výskytem výpad-ku elektřiny. Uvažuje se také o využití UPS jako zdrojů střídavého napětí i jinde.

Software

Hry

Osobně nejsem velkým vyznavačem počítačových her, ale domnívám se, že by se jich dalo využít i při ekologické výchově. Při pátráních po takových hrách jsem narazil na Lincity a Real Life. Jejich užitečnost posuďte sami.

Lincity – Účelem hry je vybudovat a starat se o město. Obyvatelům musíte zajistit potravu, bydlení, práci a majetek. Můžete buď vybudovat udržitelnou ekonomiku s využitím ener-gie z obnovitelných zdrojů a recyklace, nebo můžete spět k zániku a vybudovat si raketu, abyste unikli z planety, jejíž zdroje jsou vyčerpané a jež je znečištěná. Volba je jen na vás. Protože zdroje jsou omezené všude, nejde o hru na dlouhou dobu. Balíček obsahuje sou-bory společné pro obě verze hry (X Windows a SVGALIB).
Real Life (*http://www.sunysb.edu/philosophy/faculty/gmar/realife.html*) – Ve hře *Conway's -Game of Life* (*http://en.wikipedia.org/wiki/Convay's_Game_of_Life*) je každá z buněk buď naživu (má hodnotu 1), nebo je mrtvá (má hodnotu 0). Oproti tomu ve hře Real Life (sku-tečný život) je toto omezení nahrazeno hodnotami představujícími skutečná stádia života a smrti. Hra Real Life ukazuje závislost na počátečních podmínkách charakteristických pro neuspořádané systémy.
Společnost Sierra (*http://www.sierra.com/*) vytvořila před několika lety hry *Eco Quest 1 – Lost in Rainforest* a *Eco Quest 2 – The Search for Cetus*. Tyto hry byly určeny pro systémy MS-DOS a Windows 3.x a zaměřovaly se na mladší hráče.

- Hru *SimEarth* představila roku 1988 společnost Maxis (pro DOS, Win3.x, Mac). Jedná se o simulaci vývoje planety od formování zemské kůry po vznik a rozvoj civilizace. Je zalo-žena na teorii Gaia Jamese Lovelocka. Hra je celkem nudná a složitá, ale hrou s modely získáte mnoho cenných informací, zejména o skleníkovém efektu.

Ve hře *Balance of the Planet* od Chrise Crawforda z roku 1991 (pro DOS, Mac), ve které jste v pozici autora vládní strategie, musíte skloubit průmysl a ochranu životního prostře-dí. Hra je neuvěřitelně složitá a fádní (snad ještě více než SimEarth), ale v každém přípa-dě poučná, a pokud strávíte nějaký ten čas pročítáním manuálu, tak i zajímavá. (Verzi pro Mac si můžete stáhnou na internetové adrese *http://www.erasmatazz.com/free.html*, zdá se však, že Executor neumí zpracovat názvy souborů z archivu.)
Global Effect z roku 1992, jejímž tvůrcem je Millennium (pro DOS, Amiga), je strategická hra, ve které se pokoušíte přemoci protihráče. Přitom se navíc musíte vypořádat s nega-tivními ekologickými vlivy (z průmyslu a zbrojení) působícími na vaši populaci. Ve srov-nání se současnými „střílečkami“ je celkem nudná a grafika není příliš zajímavá.
SimIsle z roku 1995 od Maxis (pro DOSR) – rozvíjíte tropický ostrov, aniž byste narušili ekologickou stabilitu deštného pralesa. Hra je výchovná i zábavná zároveň.
SimPark z roku 1997 od Maxis (pro Win 95) – jedná se o dětskou verzi SimIsle, je jedno-dušší a více výchovně zaměřena.

Předpokládám, že starší hry pro DOS by měly v pohodě běžet na dosemu.

Modelování, sběr dat, statistika atd.

Pro systém Linux je k dispozici program *Ecolab*, ale existuje i další software (pro MS Windows), který se využívá pro ekologii. Předpokládám ale, že software Linux (například databáze nebo sta-tistické programy) lze potřebám ekologického výzkumu snadno přizpůsobit. S jeho pomocí je také možné provádět ekologické modelování.

Ecolab

Ecolab je jednak název balíčku software, ale i výzkumný projekt zabývající se dynamikou evolu-ce. EcoLab (*http://parallel.acsu.unsw.edu.au/rks/ecolab.html*) je systém, který vytváří abstraktní ekologické modely. Byl vytvořen v Tcl/Tk, takže můžete za chodu snadno měnit parametry mode-lu pomocí úpravy skriptu. Model samotný je vytvořen v C++.

OpenClassroom

OpenClassroom je určen pro vzdělávání. Na internetové adrese *http://www.openclasroom.org/* je k dispozici softwarový balíček umožňující vytvářet skupiny sdílející vědomosti připojením počíta-čů (starých, nových) do sítě, ať lokální či celosvětové. Hlavním zájmem této iniciativy je umožnit vzdělávacím a veřejným organizacím, aby prodloužily životnost svého vybavení aktuálním soft-warem, který mohou používat ve svých počítačích.

Tierra

Tierra (*http://www.nis.atr.jp/~ray/tierra/*) je nástroj pro studium evoluce a ekologie; spustitelná je na OS Linux a dalších OS.

Uplatnění systému Linux při výzkumu v oblasti životního prostředí

Se svolením Wade W. Hamptona: Systém Linux se může skvěle uplatnit při zkoumání životního prostředí. Existují řešení, která lze využít při dálkovém sledování a získávání dat. Existují VELMI malé implementace systému Linux, jako je například uCsimm (*http://www.uclinux.com/*). Systém Linux se také používal při zkoumání počasí v letadle Hurricane Hunter NOAA. Je také ideální plat-formou pro získávání ekologických a environmentálních informací v síti Internet pomocí klasic-kých webových prohlížečů, jakým je například Netscape. Lze jej rovněž využít ke komplexnímu modelování ekologických a environmentálních procesů.

SWARM

SWARM (*http://www.swarm.org/*) je softwarový balíček pro multifaktoriální modelování vyvinutý firmou Swarm Development Group (SDG). Je určen vědcům z celé řady oborů, zejména pro zkoumání v oblasti umělého života. Základní funkcí programu Swarm je modelování kolekcí souběžně působících faktorů. Poté lze vytvořit širo-kou škálu modelů, které jsou založeny na různých faktorech. Zdrojový kód je volně dostupný a jeho používání se řídí licenčními podmínkami GNU.

Změny klimatu

Projekt Climate-Dynamics (*http://www.climate-dynamics.rl.ac.uk/*) je projekt, který si klade za cíl analýzu klimatu založenou na sdílení zdrojů počítačových klientů.

UNCERT

UNCERT (*http://uncert.mines.edu/*) je softwarový balíček určený pro analýzu geostatické nerov-nováhy aplikované na modelování proudění podzemní vody a přenosu kontaminantů. Byl vyvi-nut pro vyhodnocování vnitřní nerovnováhy při popisování podpovrchové geologie, hydraulic-kých vlastností a migrace nebezpečných kontaminantů v systému podpovrchového proudění. Skvěle se hodí k výše uvedeným účelům, je však i dostatečně všestranný, a mohou jej proto vyu-žívat i vědci z různých oblastí.

EcoTopia

Webová stránka EcoTopia (*http://www.ecotopia.org/*) využívá počítačovou simulaci k modelování města Santa Cruz ve státě Kalifornie jako vzorové komunity šetrné k životnímu prostředí. Ekotu-ristům a vyznavačům „zeleného“ způsobu života se EcoTopia pokouší pomocí počítačového modelování a nastíněním prognóz nabídnout model slučující technologie a snižování zátěže život-ního prostředí.

Digiqual

Digiqual (*http://sourceforge.net/projects/digiqual/*) je balíček programů pro řízení systému kontro-ly životního prostředí a kvality v továrně. Balíček se skládá ze šesti modulů, z nichž pouze první byl zatím uveden na trh. Jedná se o moduly Supplier Evaluation, Non Conformity, Documentati-on, Maintenance, Internal Audit a Training. Všechna data se prostřednictvím konektoru psycopg ukládají do databáze PostgreSQL.

Různé další informace

Související projekty, seznam adres a diskusní skupiny

Repair FAQ (*http://www.repairfaq.org/*).

- *386 World* (*http://come.to/386*), Gaute Hvoslef Kvalnes <*386@altavista.net*>. Jedná se o jeden z nejlepších dostupných zdrojů o počítačích „386“ kompatibilních a o příslušném softwaru. Autor se zabývá spíše operačním systémem MS Windows, ale podporuje také Linux.

Electronic Green Journal (*http://www.lib.uidaho.edu/*) vydávaný knihovnou University v Idaho je profesionální, odborná publikace, která se věnuje tématům ochrany životního prostředí včetně hodnocení stavu životního prostředí, ochrany přírody, rozvoje, ukládání odpadů, ekologické výchovy, rizik, znečištění, zdrojů, technologií a úpravy. Na tento pro-jekt přispívá akademická obec, záměrem je však vydávat články, bibliografie, posudky

> a oznámení jak pro zainteresovanou veřejnost, tak i pro odborníky. Uvítají autorské pří-spěvky na jakékoliv z výše vedených témat.

BAN je globální síť organizací zabývajících se rozvojem a znečištěním, které ctí myšlenku mezinárodního práva v oblasti životního prostředí. Hledají způsoby, jak předcházet jaké-mukoliv obchodu se znečištěním – toxickým odpadem, toxickými výrobky a technologie-mi. Členové BAN budou pracovat na národní, regionální a globální úrovni za účelem dosa-žení následujících cílů: Posláním Basel Action Network (BAN – *http://www.ban.org/*) je předcházet globalizaci problematiky toxických chemikálií. Především se chceme ujistit, že Basilejská úmluva a dle ní zakázaný vývoz nebezpečných odpadů z členských států OECD do nečlenských států nebude zeslabovat, ale dojde co nejdříve k jeho ratifikaci a imple-mentaci. Také chceme zajistit, že Basilejská úmluva a další nástroje slouží k zamezení obchodu a růstu nejnebezpečnějších a často zastaralých průmyslových odvětví, zejména s ohledem k rozvojovým a nově se industrializujícím státům.

Silicon Valley Toxics Coalition (SVTC – *http://www.svtc.org/*) je různorodá koalice, která se již téměř dvacet let angažuje ve výzkumu, advokacii a zabývá se problémy životního pro-středí a lidského zdraví způsobenými prudkým nárůstem vyspělého elektronického prů-myslu. Naším cílem je zvýšení udržitelnosti životního prostředí, čistá produkce průmyslu, zlepšení zdraví, prosazování práva a zabezpečení demokratického rozhodování pro obce a pracovníky, jejichž životy jsou bezprostředně ovlivněny průmyslem, jako je tomu napří-klad v Silicon Valley a podobných oblastech ve Spojených státech a ve světě.

Ekologické srovnání počítačů

Vědci z Technické univerzity v Berlíně (Technical University of Berlin – *http://tu-berlin.de/*) řešící projekt zabývající se znovupoužitím počítačů (projekt ReUse – *http://www.reuse-computer.de/*) nedávno porovnali spotřebu energie různých typů počítačů během jejich životnosti. K výrobě počítačů je zapotřebí výkonu 535 kW/h, což je o deset procent méně než před čtyřmi lety. Větši-na energie se spotřebuje během osmihodinové pracovní doby. Spotřeba energie nových počítačů s procesory o frekvenci 2,5–3 GHz je 100 wattů, i když jsou v pohotovostním – úsporném režimu, zatímco počítače s procesorem 1,4 GHz spotřebují 80 wattů a 4 roky staré počítače spotřebovaly pouze 60 W. Z ekologického hlediska je proto lepší koupit si starý počítač jednak proto, že se nespotřebovává energie k výrobě nového, a také proto, že jeho provoz je méně energeticky náročný.

Obrazovky LCD spotřebovávají méně energie než jakékoliv jiné monitory. Z tohoto důvodu jsou nejekologičtějšími typy počítačů notebooky. Během provozu spotřebovávají nejméně energie. Tři roky staré laptopy jsou z tohoto hlediska lepší, protože jejich procesory nejsou tak energeticky náročné jako nové modely. Článek na toto téma naleznete v německém počítačovém časopise C't 21/2003 (*http://heise.de/ct/*).

Služby PCMCIA Card Services a vylepšené řízení spotřeby

Převzato z dokumentu PCMCIA-HOWTO (*http://tldp.org/HOWTO/PCMCIA-HOWTO.html*): „Pokud jste nakonfigurovali jádro s podporou APM, lze služby PCMCIA Card Services kompilovat s pod-porou pro vylepšené řízení spotřeby (APM). Jestliže je ve vašem systému rozpoznána kompatibil-ní verze, moduly PCMCIA se pro APM nakonfigurují automaticky. Ať již je podpora APM nastave-na nebo není, pro vypnutí a restartování karet PCMCIA můžete před uvedením notebooku do reži-mu spánku zadat příkaz cardctl suspend a po opětovném spuštění příkaz cardctl resume. S aktivním modemem nebude tento postup fungovat, protože sériový ovladač není schopen ulo-žit a obnovit operační parametry modemu. Zdá se, že APM je v některých systémech nestabil-ní. Pokud dojde ve vašem systému k nějakým potížím s APM a PCMCIA, pokuste se zjistit, který z balíčků je příčinou, a teprve poté chybu oznamte. Některé ovladače (především pak ovladače PCMCIA SCSI) se z cyklu režim spánku/obnovení nedokážou zotavit. Používáte-li kartu PCMCIA, zadejte před uvedením systému do režimu spánku vždy příkaz cardctl eject.

Pokud je to možné, používejte místo modemu PCMCIA interní modem (může to být WinModem).“

Metody úspory energie

Nepotřebujete-li podporu infračerveného zařízení, zakažte ji v systému BIOS nebo vypněte ovladač zařízení IrDA. Některé funkce IrDA jádra jsou pro úsporu energie užitečné. V tech-nických vlastnostech k notebooku HP OmniBook 800 se například doporučuje vypínat infračervený port, pokud není zrovna používán, protože může spotřebovat až 10 procent pohotovostního času baterie. Můžete také zkusit zakázat funkci *Fast RRs* v sekci IrDA jádra. Díky této možnosti získáte mnohem lepší dobu odezvy, ale spotřebujete více energie.

Služby PCMCIA spotřebovávají mnoho energie, takže pokud je nepoužíváte, raději je vypněte.

Návrhy, jak zvýšit pracovní dobu baterie až na 8 hodin, naleznete na webu Adorable Tos-hiba Libretto (*http://www.silverace.com/libretto/*).

Informaci o APM naleznete výše v části o Vylepšeném řízení spotřeby (APM/ACPI).

Upravený nástroj rclock (*http://www.leland.stanford.edu/~bbense/toys/*). Booker C. Bense upravil nástroj rclock, aby mohl do rozhraní hodin přidat jednoduchý měřič energie baterie.

xbatstat (*http://www.jaist.ac.jp/~daisuke/Linux/xbatstat.html*) – zkoušeč stavu baterie pro Linux a X.

- KDE (*http://www.kde.org/*) nabízí nástroje *KAPM*, *Kbatmon*, *Klaptop* a *KPowerSave*. Auto-rem je Paul Campbell. *Kcmlaptop* je sada ovládacích panelů KDE, která implementuje pod-půrné funkce do notebooků, včetně sledování stavu baterie – stručně řečeno se jedná o malou ikonu ve stavovém řádku KDE, která ukazuje, kolik času zbývá do vybití baterie. Varuje vás v případě, že se sníží stav baterie, a umožní vám nastavit možnosti pro úsporu energie. Podobné softwarové balíčky naleznete rovněž na webu projektu GNOME (*http://www.gnome.org/*).

Více informací získáte na webu Battery-Powered-mini-HOWTO (*http://tldp.org/HOWTO/ Battery-Powered/index.html*).

Toshiba fan – zapnutí nebo vypnutí ventilátoru v laptopu Toshiba Pentium. Jedná se

> o nástroj příkazového řádku k vypnutí a zapnutí ventilátoru nebo ke zjištění aktuálního stavu. Měl by fungovat na všech laptopech Toshiba Pentium, které mají ventilátor.

Hardwarový přehrávač MP3

V německém počítačovém časopise CT (*http://www.heise.de/ct/*) je na straně 200 čísla 9/1999 a na straně 260 čísla 10/1999 článek o tom, jak ze starých počítačů (řady 286 a novějších) vytvořit pře-hrávače MP3 prostřednictvím dekodéru MP3-Hardware-Dekoder na paralelním portu. (Autorovu domovskou stránku naleznete na internetové adrese *http://www.mp3pump.de/english/index.html.*)

Na zmíněném webu je také informace o softwaru od Klause Peichla (*http://leute.server.de/ peichl/mpegcd.htm*), který nevyžaduje použití dekodéru hardwaru. Oba zmíněné programy jsou založeny na systému DOS.

Program *Cajun* (*http://www.cajun.nu/*) vám umožňuje vytvořit z jakéhokoli počítače (>75 MHz) audio jukebox pro své auto nebo domácnost. Využívá sériový displej MatrixOrbital (*http://www.matrixorbital.com/*) a podporuje rozhraní IRman pro dálkové ovládání. Výstup ze zvukové karty směřuje do vašeho zesilovače (ať již v autě nebo doma). Nabízí podporu FM/Video4Linux, CrystalFontz a výběr mezi mpg123 nebo xaudio. Otázkou pouze zůstává, kolik energie spotřebuje takový přehrávač oproti těm běžně dostupným.

Na co si dát pozor při koupi nového počítače

Se svolením Wade W. Hamptona (úpravy provedl wh): Kupte si energeticky nenáročný počítač, jakým je laptop nebo síťový počítač. Ty nespotřebují tolik energie jako stolní počítačové systémy. Na webu jsem se dočetl, že někdo používal Corel/Rebel Netwinder se solárními bateriemi. Je zvláštní, že počítač s hodnocením „Energy Star“ potřebuje 300wattový zdroj energie a spotřebuje mnohem více energie než počítač, jakým je Netwinder, kterému stačí pouhý 10wattový zdroj (a přesto vyhovuje hodnocení počítačového vybavení Energy Star, které se zaměřilo na spotřebu nevyužité energie).

Možná by měla existovat třída počítačů nazvaná například „Energetický lakomec“ (nebo podob
ně), protože spotřebovává řádově méně energie než systémy s hodnocením Energy Star. Pořízením monitoru LCD namísto klasického monitoru CRT ušetříte energii. Monitory LCD spotře-bovávají pouze 30–40 wattů, zatímco většina ostatních monitorů spotřebuje stovky wattů. Cena monitorů LCD je stále vyšší než u jiných typů, ale rozdíly se rapidně snižují.

Ujistěte se, že nově koupený počítač obsahuje hardware s podporou ACPI (APM) a má nízké záření. Používejte monitory vyhovující hodnocení TCO, DPMS nebo Energy Star.

R. Horn <*rjh@world.std.com*> napsal: „Web Lawrence Berkeley Labs – LBL (*http://eande.lbl.gov/*) osob-ně považuji za nejlepší zdroj informací, co se týče energeticky úsporného vybavení. Zabývá se důleži-tými podrobnostmi snížení energetické spotřeby mnoha různých druhů zařízení, nejen počítačů. Nabí-zí také mnoho odkazů na související weby. Hodnocení Energy Star je projekt americké agentury pro ochranu životního prostředí (US Environmental Protection Agency). Doposud se všechna omezení vycházející z tohoto projektu zaměřovala na snížení spotřeby energie bez nutnosti změnit nebo ome-zit běžné užívání. Množství elektrické energie spotřebované vypnutými zařízeními (počítače, televize, mikrovlnné trouby atd.) je až neuvěřitelné. Velké množství energie rovněž zbytečně spotřebují zaříze-ní, která musí zůstávat zapnuta (vyznačení nouzových východů, dopravní světla apod.). Na webové stránce LBL naleznete údaje o skutečné spotřebě energie různých počítačů. Údaj o 300wattovém zdro-ji energie je poněkud zavádějící. Skutečná spotřeba energie se liší v závislosti na tom, jaké programy jsou spuštěné a jestli lze vypnout pevný disk. Skutečná spotřeba se běžně pohybuje v rozmezí 50–75 wattů. Při nečinnosti systému spotřeba významně klesá.

NetWinder je pěkný počítač, který však vyžaduje funkční kompromisy. Výkon procesoru při největším výkonu je o hodně nižší. Operační systém není Windows. Má i další omezení. Mohli bychom jej porovnat s typický laptopem. Obvykle mohou počítače tohoto typu čerpat energii z drobného solárního panelu, protože průměrná energetická spotřeba je poměrně nízká. Na těchto počítačích lze porovnávat cenu ve vztahu se spotřebou energie. Mají stejný výkon jako klasické počítače, ale nízká spotřeba energie zdvojnásobuje, či dokonce ztrojnásobuje cenu.

(Osobně používám Psion. Dobrý, ale pomalý počítač, kterému stačí pouze 200 mW. Mohl bych do
něj nainstalovat dokonce i systém Linux, pokud by se vyřešily některé problémy s pamětí ROM.) Rozsáhlá debata na téma zavedení omezení do hodnocení Energy Star řešila, co by mělo být cel-kově výhodnější – zda drobná a cenově zanedbatelná vylepšení veškerého prodávaného vybave-ní, nebo mnohem dražší, zato značná zdokonalení. Daly by se tyto peníze investovat někde jinde
s větším přínosem? Jak budou nakupující reagovat na vyšší ceny? Shoda zatím panuje v tom, že zdokonalení velkého počtu počítačů při nepatrném zvýšení cen je moudřejší než vylepšení malého počtu počítačů za vysokou cenu.“

Hardware šetrný k životnímu prostředí

Se svolením Wade W. Hamptona a Knuta Sueberta: Nový typ hardwaru, na kterém se má instalo-vat systém Linux, by měl využívat technologie šetrné k životnímu prostředí, například nízkoener-getické procesory (jako je ARM Intel), úsporné baterie, nízkoenergetické monitory (například LCD bez podsvícení), menší obaly atd. Linux podporuje širokou škálu hardwaru a technologií. Ty se mohou stát výkonným, flexibilním a navíc k životnímu prostředí šetrným řešením založeným na systému Linux. Mělo by se zavést hodnocení dopadu nového hardwaru pro systém Linux na život-ní prostředí. Hodnocen by mohl být i určitý software založený na systému Linux, jako je například bottlerocket (X10). Zařízení, jako je Netwinder nebo uCsimm, by

dosáhla vysokého hodnocení díky své velikosti, nízké spotřebě energie, kapacitě atd. V německém časopise Telepolis (*http://www.telepolis.de/tp/deutsch/inhalt/te/1367/1.html*) vyšel článek o plýtvání zdrojů během výroby počítačů. Obecně vzato, procesory PPC spotřebovávají méně energie než procesory řad x86.

Ekologické označení pro počítače

Známá jsou například označení TCO, DPMS, Energy Star Blauer Engel (*http://www.blauer-engel.de/*) nebo GEA (Group for Efficient Appliances). Zjistěte si, co znamenají, a řiďte se jimi při koupi počí-tače a příslušenství.

Jiné operační systémy DOS

Existuje ještě mnoho nástrojů, díky nimž mohou být počítače s procesorem řady 286 stále užiteč-né. Na webu Simtel.Net (*http://www.simtel.net/*) se po zadání klíčového slova „286“ zobrazí mnoho odkazů na zajímavý shareware. Například:

Lokální síť (LAN) pro počítače MSDOS, 286+req (ftp://ftp.simtel.net/pub/simtelnet/msdos/ lan/neos-10.zip)
RoseMail, PCBoard, 8086/286 exe (ftp://ftp.simtel.net/pub/simtelnet/msdos/pcboard/ rm172b.zip)
Prostředí pro paralelní zpracování úloh a pro více uživatelů v počítačích řady 286–586
(ftp://ftp.simtel.net/pub/simtelnet/msdos/sysutl/vmix285.zip)

Společnosti zabývající se recyklací

Zjistěte si adresy společností, které se zabývají recyklací odpadu, a začněte využívat jejich služeb. Dobrým startem může být stránka *http://www.firmy.cz/Remesla-a-sluzby/Ekologicke-sluzby/ Recyklace-odpadu.*

Webkamera v Linuxu

Úvod

Tento dokument by měl čtenáři pomoci při nastavení a konfiguraci webkamery, digitální kamery nebo jiného videozařízení v operačním systému Linux. Vysvětluje, jak povolit nezbytnou podpo-ru jádra a/nebo softwaru a různé aplikace pro zachycování snímků, které lze používat pro přístup k zařízení. Nezabývá se rozdíly v grafických a videoformátech, funkcemi a/nebo schopnostmi kon-krétních zařízení ani kódováním či konverzí formátu videa.

Povolení podpory pro hardware (webkameru) v Linuxu

Ovladače a moduly

Aby webkamera fungovala, musíte zajistit podporu pro připojení a pro vlastní hardware kamery. Ti, kteří se již vyznají v jádrech a modulech a vědí, jak je zavést, by měli přejít k části 2.2, která řeší podporu typu připojení. Pokud víte, že je vaše sběrnice USB, IEEE 1394 nebo jiná sběrnice připojovaná ke kameře již nakonfigurovaná a funguje, měli byste přejít k seznamu konkrétního hardwaru webkamery uvedenému v části 2.3.

Ovladače pro webkameru jsou obvykle k dispozici jedním ze tří způsobů: v rámci jádra, jako kom-pilovatelný samostatný modul nebo jako předkompilovaný (zabalený) binární ovladač z distri-bučního balíčku Linuxu.

Modul či jádro?

Zpravidla bude potřebná podpora k dispozici v jádru (stock kernel) či pracující části operačního systému ve výchozí instalaci. Váš dodavatel distribučního balíčku Linuxu již pravděpodobně povo-lil většinu obvykle používaných možností, včetně sběrnice či typu připojení a ovladačů pro běžné modely kamer. Ovladač je k dispozici buď jako modul pro zavedení nebo v rámci již používané-ho jádra.

Zda je ovladač již povolen, zjistíte snadno prostřednictvím příkazu dmesg ve spojení s příkazem less (pro snadné stránkování), s jehož pomocí vyhledáte potvrzení, že došlo k zavedení ovlada-če při spouštění systému:

```
$ dmesg | less
```

...výsledkem může být v závislosti na vašem hardwaru například následující výpis:

```
Dec 18 17:35:18 localhost kernel: hub 5-0:1.0: USB hub found Dec 18 17:35:18 localhost kernel: hub 5-0:1.0: 2 ports detected Dec 18 17:35:18
localhost kernel: Linux video capture interface: v1.00 Dec 18 17:35:18 localhost kernel: quickcam: QuickCam USB camera found
(driver version QuickCam USB $Date: 2005/01/07 13:29:53 $) Dec 18 17:35:18 localhost kernel: quickcam: Kernel:2.6.7 bus:1 class:FF
subclass:FF vendor:046D product:0840 Dec 18 17:35:18 localhost kernel: quickcam: Sensor HDCS-1000/1100 detected Dec 18 17:35:18
    localhost kernel: quickcam: Registered device: /dev/video0 Dec 18 17:35:18 localhost kernel: usbcore: registered new driver quickcam
```

Pokud se podobný výpis nezobrazí, konkrétní ovladač může existovat jako zaváděný modul. Znáte-li název modulu, zkuste použít příkaz find; v tomto příkladu hledáme modul ibmcam:

```
$ find /lib/modules -name ibmcam.o
```

Moduly měly až do verze 2.4 příponu .o, kterou u jader verze 2.6+ nahradila přípona .ko. Seznam všech dostupných modulů získáte za použití těchto příkazů v příkazového řádku:

```
$ ls -R /lib/modules/`uname -r`/kernel
```

Položka `uname -r` obklopená obrácenými apostrofy uvádí číslo verze vašeho jádra. Následující výpis je příklad toho, co můžete zjistit v jádru připraveném na používání webkamery se sběrnicí USB, přičemž se vše zavede jako modul (pro stručnost uvádíme pouze relevantní řádky):

```
./usb: usbvideo.o usbcore.o ibmcam.o
```

Když víte, jaký modul vaše kamera potřebuje, můžete pomocí následujícího příkazu zjistit, zda je příslušný modul již zaveden:

```
# lsmod
```

Jak ukazuje výzva uvedeného příkazu, budete k této činnosti potřebovat oprávnění root. Výstup

by měl vypadat zhruba takto:

```
cdrom                   29312   0   (autoclean) [sr_mod]
usb-ohci                17888   0   (unused)
usbcore                 56768   0   [scanner ibmcam usbvideo usb-ohci]
ibmcam                  39680   0
```

Většina jader se kompiluje pomocí direktivy kmod, což umožňuje automatické zavedení potřebných modulů při detekci příslušného hardwaru. Avšak nemusí tomu tak být vždy, takže pokud jste neza-vedli konkrétní modul a myslíte si, že by mohl být k dispozici, zkuste jej zavést ručně prostřednic-tvím příkazu modprobe, jako v následujícím příkladu, kde jsme použili modul ibmcam:

```
# modprobe -v ibmcam
```

V části „Modely webkamer“ naleznete ovladače pro konkrétní modely webkamer, případně odka-zy na stránky s kódy pro ovladače. Ovladače jsou obvykle dostupné jedním z těchto třech způ-sobů: v rámci jádra, jako kompilovatelný samostatný modul nebo jako předkompilovaný binární ovladač v distribučním balíčku Linuxu.

Jestliže podpora pro váš ovladač není povolena staticky v rámci jádra nebo jako modul, nezou-fejte. Ve zdroji jádra systému Linux jsou k dispozici ovladače pro mnoho modelů (přímo v repo-zitáři zdrojového kódu na *http://www.kernel.org*) nebo v kódu nabízeném samostatně, který lze konfigurovat pro práci s vaší aktuální instalací, jak popisuje kapitola „Patch, zdroj či předkompi-lovaný binární ovladač?“. Je-li ovladač pro vaši webkameru dostupný ve zdroji jádra, přitom však není povolen jako modul ve vašem výchozím systému, můžete buď znovu kompilovat jádro ze zdrojového kódu, který máte k dispozici, nebo získat novou verzi zdroje jádra, ať již připravenou vaším distributorem Linuxu či přímo z výše uvedeného odkazu (jako tzv. jádro „vanilla“). Jestliže neznáte požadavky a procedury při kompilaci vlastního jádra, získáte potřebné informace na inter-netové adrese *http://www.tldp.org/HOWTO/Kernel-HOWTO.html*.

Patch, zdroj či předkompilovaný binární ovladač?

Možná zjistíte, že vaši webkameru podporuje pouze patch jádra; anebo ovladač, který je součástí zdroje a nevyžaduje překompilování jádra; nebo jste měli štěstí a váš distribuční balíček nabízí předkompilovaný binární ovladač pro strukturu vašeho počítače. Postup u prvních dvou možnos-tí sahá daleko za možnosti této knihy a pravděpodobně nejlépe jej popisuje dokumentace, kterou naleznete na webové stránce ovladače konkrétního modelu (jejich přehled viz kapitola „Modely webkamer“). Obecnější informace o těchto procesech však nabízí kapitola „Řešení potíží“.

Podpora typu připojení Webkamery USB

Máte-li webkameru USB, je pravděpodobné, že pro dané zařízení byl vytvořen linuxový ovladač. V systému Linux existují dva způsoby podpory zařízení USB. Jedním z nich je tradiční podpora jádra a druhým je libusb. Nejméně pro jednu kategorii

webkamer – modely založené na STV0680
– je doporučována podpora libusb, alespoň podle webové stránky Sourceforge (*http://stv0680-usb.sourceforge.net/*).Jestliže nevíte, zda váš ovladač vyžaduje podporu libusb, měli byste pravděpodobně zůstat u běž-nější podpory jádra pro zařízení USB, které se budeme věnovat v kapitole „Podpora USB v jádrusystému Linux“.

Libusb

Libusb je knihovna, která umožňuje přístup k funkcím USB v systému Linux prostřednictvím uži-vatelského prostoru (userspace) a bez nutnosti povolení podpory jádra a vkládání modulů. Větši-na distribučních balíčků nabízí knihovnu libusb (*http://libusb.sourceforge.net/*) ve své stabilní verzi (a některé je implicitně instalují), takže pokud již nemáte podporu jádra pro zařízení USB, může-te jen nainstalovat balíček libusb a zajistit tak, aby vaše zařízení fungovalo. V jádru musí být povo-lena podpora systému souborů pro zařízení USB, což zajišťuje většina distribučních balíčků. Abys-te se ujistili, spusťte v příkazovém řádku následující příkaz:

```
$ cat /proc/filesystems
```

Kromě jiného byste měli vidět toto:

```
nodev usbdevfs nodev usbfs
```

Možná bude nutné použít příkaz mount pro povolení podpory usbdevfs a zobrazení souborů zařízení. Můžete tak učinit v příkazovém řádku zápisem mount -t usbdevfs none /proc/bus/usb. Nezkoušejte použít knihovnu libusb, pokud je pro webkameru povolena pod-pora *jádra*, ať již staticky nebo zavedením modulu; používat lze pouze jednu z těchto možností.

Balíček libusb získáte ve formátu .rpm, .tgz či .deb ze svého distribučního kanálu systému Linux.
Podpora USB v jádru systému Linux

Jestliže nepoužijete knihovnu libusb (popsanou výše), je pro webkameru typu USB nezbytná pod
pora jádra.U jader verzí 2.2 a 2.4 může vaše webkamera USB vyžadovat funkci usbvideo. Tato funkce nenívyžadována ve verzích 2.6+.

Pro obecnou podporu sběrnice USB v systému Linux budete potřebovat podporu podsystému USBv jádru, ať již usb-ohci, usb-ehci či jakýkoli typ ovladače USB, který váš systém upřednostňuje.Podpora podsystému USB je v jádru systému Linux k dispozici od novějších verzí 2.2. Podrobněj-ší informace o podpoře USB naleznete na webu *http://www.linux-usb.org/*. Jestliže chcete zjistit,jaké moduly jsou zavedeny, napište do příkazového řádku nebo terminálu xterm následující:

```
# lsmod
```

Jak ukazuje výzva uvedeného příkazu, musíte mít k této činnosti oprávnění root. Měli byste zís

kat výstup podobný tomuto:

```
cdrom                   29312   0   (autoclean) [sr_mod]
usb-ohci                17888   0   (unused)
usbcore                 56768   0   [scanner ibmcam usbvideo usb-ohci]
ibmcam                  39680   0
```

Není-li zaveden konkrétní hledaný model, přitom však předpokládáte, že by mohl být dostupný, zkuste jej zavést přímo (jako příklad jsme použili usb modul ibmcam):

```
# modprobe -v ibmcam
```

Po provedení tohoto příkazu by se měl zobrazit výpis podobný tomuto:

```
Using /lib/modules/2.4.20/kernel/drivers/usb/ibmcam.o
```

Umístěním položky ibmcam (například) do /etc/modules (může se lišit podle typu distribuce, může to být např. /etc/modprobe.conf) můžete zajistit automatické zavedení modulu při spouš-tění. Zavedení modulu si můžete potvrdit kontrolou syslogu nebo v záznamu průběhu spouštění pomocí dmesg | less, v kterém byste měli vidět výpis podobný tomuto:

```
Oct 18 12:43:12 K7 kernel: hub.c: new USB device 00:02.3-2, assigned address 3
    Oct 18 12:43:12 K7 kernel: ibmcam.c: IBM PC Camera USB camera found (model 2, rev. 0x030a)
    Oct 18 12:43:12 K7 kernel: usbvideo.c: ibmcam on /dev/video1: canvas=352x240 videosize=352x240
```

IEEE 1394 (Firewire™, i.Link™)

Webkamery IEEE 1394 vyžadují kartu PCI IEEE 1394 nebo port sběrnice IEEE 1394 na základní desce. Rozhraní IEEE je v systému Linux podporováno od prvních verzí jádra 2.4. Máte-li štěstí a vlastníte takové zařízení, najdete informace o podpoře sběrnice IEEE 1394 v Linuxu na interne-tové adrese *www.linux1394.org*. Pokud je verze jádra starší než 2.4.2, budete muset použít jeden z patchů na webu *http://download.sourceforge.net/linux1394* odpovídající vaší verzi jádra. Navíc budete potřebovat i *http://download.sourceforge.net/libraw1394*. Na výše uvedeném webu linux1394.org naleznete i skvělého průvodce instalací.

Přehled funkcí kamer IEEE 1394 spolu s aktuálním stavem podpory pro jednotlivé moduly nalez-nete na internetové adrese *http://www.tele.ucl.ac.be/PEOPLE/DOUXCHAMPS/ieee1394/cameras/* (seznam vytvořil Damien Douxchamps).

Podpora pro webkamery na paralelní port

U verzí 2.2 a 2.4 musí být podpora paralelních portů povolena staticky nebo jako modul (u vesta-věných jader je tato možnost obvykle implicitně povolena). Možná si budete chtít nejdříve přečíst obecnější informace na internetové adrese *http://www.torque.net/linux-pp.html* o podpoře zaříze-ní s paralelními porty v rámci jádra Linux. Abyste ověřili, zda je zaveden modul parport, může-te zkontrolovat soubor dmesg nebo použít příkaz lsmod, uvedený výše. Po použití příkazu dmesg | less byste měli vidět (spolu s mnoha dalšími řádky) následující:

```
Mar 3 08:00:25 K7 kernel: parport0: PC-style at 0x378 (0x778) [PCSPP,TRISTATE] Mar 3 08:00:25 K7 kernel: parport0: irq 7 detected
```

Pokud kompilujete vlastní jádro, povolte podporu Parallel Port. Měli byste povolit režimy přeno
sů IEEE 1284, a pokud máte počítač x86, měli byste povolit také hardware typu PC. Jestliže přezkoušení režimu vrátí chybu, když se pokusíte zavést modul, možná je při vyvolání pří-kazu modprobe nutné zjistit a doplnit hardwarovou adresu. Nejčastější adresou je 0x378 pro systém x86; dalšími možnostmi jsou 0x278 a 0x3BC pro integrované nebo paralelní porty ISA. Pří-davné paralelní porty PCI mohou mít neobvyklé základní adresy. Možné je také používat více zaří-zení s moduly parport_pc či parport_arc, toto téma je však nad rámec tohoto návodu.

> Varování
>
> Před zadáním informací do příkazového řádku se ujistěte, že máte správnou adresu, jinak by mohlo dojít k nestabilitě počítače, k jeho selhání nebo k jiným potížím.

Paralelní port by měl být nastaven na režim EPP, případně ECP/EPP. Vyhovovat by mohl i obou-směrný režim (označovaný také jako „BPP“ či „PS/2“), který je však mnohem pomalejší. „Jedno-směrný“ režim není vhodný pro skenování. K výše uvedenému nastavení můžete obvykle přistu-povat prostřednictvím nabídky BIOS, alespoň co se týče systémů x86.

Modely webkamer

Nezapomeňte, že uvedené informace se budou často měnit. Používáte-li webkameru USB, měli byste navštívit web *http://www.qbik.ch/usb/devices/index.php*, případně *http://www.exploits.org/v4l/*. Níže uvedené informace týkající se konkrétních modelů webkamer pocházejí ze stejného zdroje, takže na uvedeném webu získáte jejich aktuálnější verzi. Pokud v seznamu není uveden váš hardware, může-te použít odkazy na zdroje a podle nich vytvořit svůj vlastní ovladač!

> Doporučení
>
> Pokud vaše kamera není v seznamu uvedena a chcete zjistit, zda je podporována, nejjed nodušší je ověřit u jejího výrobce, jaká čipová sada byla použita.

Tyto informace jsou obvykle k dispozici ve specifikacích uvedených v příručce k vaší webkame

ře nebo na webu výrobce. Nemůžete-li najít model své webkamery a nejste si jisti, jaká čipová sada je v ní použita, měli bysteprohledat web *https://listman.redhat.com/mailman/listinfo/video4linux-list* a/nebo se přihlásit k odbě-ru aktuálních informací.

Digitální webkamera 3 com HomeConnect

Tento model je podporován patchem jádra, který naleznete na internetové adrese *http:// homeconnectusb.sourceforge.net/*. Po instalaci patche může být v závislosti na verzi jádra nutné překompilovat jádro.

Webkamery založené na CPiA

Aktuální informace naleznete na internetové adrese *http://webcam.sourceforge.net/*. Tato čipová sada se používala při výrobě webkamer typu USB i s paralelním portem, včetně těchto:

Aiptek HyperVcam Fun USB (jiné než typu OV511)
Creative Video Blaster WebCam II USB a paralelní port
CVideo-Mail Express paralelní port
Digicom Galileo USB a Digicom Galileo Plus
Dynalink Digital Camera
Ezonics EZCam (ne Pro či Plus)
I-View NetView NV200M
Microtek EyeStar USB
Pace Color Video Camera USB
SuperCam WonderEye
TCE Netcam 310 USB
TerraCam USB (jiné než typu OV511 či TerraCam Pro)
Trust SpaceC@m Lite USB a SpaceC@m 100

Utopia USB Camera
ZoomCam USB a paralelní port

Webkamery USB založené na SE401, SE402 a EP800

Na tomto projektu se stále pracuje. Ovladače a další užitečné informace jsou k dispozici na webu projektu (*http://members.brabant.chello.nl/~j.vreeken/se401*). V době, kdy vznikal tento doku-ment, bylo pro získání podpory pro tyto modely nutné nainstalovat patch a překompilovat jádro. Ovladač podporuje následující:

Čipová sada SE401 prostřednictvím ovladače se401:

Aox SE401
Philips PCVC665 USB VGA webkamera „Vesta Fun“
Kensington VideoCAM PC Camera (Modely 67014-67017)

Čipové sady SE402 a EP 800 prostřednictvím ovladače epcam:

Spypen Actor
Rimax Slim Multicam
Concord Eye-Q Easy
Creative PD1001
Chicony DC-100
Endpoints SE402 a EP800

Webkamery založené na OmniVision

Do této kategorie patří množství webkamer a zařízení pro nahrávání videa vyrobených společností Omnivision, včetně OV511(+), OV518(+), OV6620, OV6630, OV7610 a OV7620AE. Tímto projek-tem se zabývá web *http://alpha.dyndns.org/ov511*. Podporovanými modely jsou:

Aiptek HyperVcam Home a Mobile
Amitech AWK-300
I-view NetView NV300M
TEVion MD9308
Intel Me2Cam
Dlink DSB C100, C300
Hawking Tech. UC-110, UC-300 a UC-310
Puretek PT-6007
Alpha Vision Tech AlphaCam SE model AC-520
Creative Labs WebCam model PD1001 s čipovou sadou OV518
Creative Labs WebCam 3, WebCam Go, WebCam Go Plus
Elecom UCAM-C1C20
Elta WEBCam 8211 PCC
Ezonics EZPhone Cam
Philips ToUCam XS (starší verze s čipovou sadou OV518)
LG Electronics LPC-UM10
Lifeview různé modely USB Life TV
Genius VideoCam Express
AverMedia Intercam Elite
Maxxtro Cam22U
MediaForte MV300, PC Vision 300
Terratec TerraCam PRO a některé modely TerraCam
OmniVision (kromě modelů s čipovou sadou OV519)
TRENDNet TV-PC301
Trust Sp@ceC@m USB
Lifetec LT9388
BestBuy EasyCam U
Maxell Maxcam
TCE NetCam 310u
Medion MD9388
Webeye 2000B
Suma eON
Prochips PCA-3100
Ezonics EZ USB Cam II (modely OV511+)
Waytech I-Pac VIC-30
Zoom Telephonics ZoomCam III USB (model 1598)

Podpora webkamer Logitech (dříve Connectix) QuickCam

Webkamery QuickCam VC typu USB a s paralelním portem podporuje ovladač nabízený na webu *http://digilander.libero.it/demarchidaniele/qcamvc/quickcam-vc.html*. Pro podporu tohoto mode-lu je nutné nainstalovat patch a překompilovat jádro.

Ovladačem QuickCam se zabývají dva různé projekty nabízející dva druhy ovladače pro určité modely QuickCam. Jedná se o samostatné ovladače, které nevyžadují instalaci patche či překom
pilování jádra. Ovladače qce-ga (*http://qce-ga.sourceforge.net/*) a qc-usb (*http://www.ee.oulu.fi/ ~tuukkat/quickcam/quickcam.html*) podporují následující modely:

Logitech (starší modely) QuickCam Express
QuickCam Web
LegoCam
Dexxa WebCam
Labtec WebCam Ovladač qc-usb je spíše experimentální, ale údajně má být vhodnější pro určité modely, jako je například QuickCam Web. Dozvěděl jsem se také, že novější verze modelu Logitech QuickCam

Express již s výše uvedenými ovladači nefungují; na internetové adrese *http://home.tiscali.dk/tomasgc/labtec/* naleznete pokusný ovladač, který by měl novější model podporovat.Poznámka pro uživatele Redhat: Ovladač qce-ga se nezkompiluje správně při použití uprave-
ného zdroje jádra ve verzi Redhat 9, opravu však naleznete na internetové adrese *http://www.ee.oulu.fi/~tuukkat/quickcam/FAQ.*

Některé modely Logitech podporuje ovladač Philips uvedený v kapitole „Webkamery Philips na USB“.

Webkamery založené na čipové sadě ICM532

Jeden z ovladačů pro tuto čipovou sadu (*http://icm532.sourceforge.net/home.html*) je nyní začle-něn do zdroje jádra verze 2.6; druhý je experimentální (dle vlastního vyjádření vývojáře) a je k dis-pozici na webu *http://home.tiscali.dk/tomasgc/labtec/*. Ovladače by měly podporovat tyto modely:

IC-Media Corp PenCam
Novější verze Logitech QuickCam Express
Novější verze Labtec WebCam
Mikroskop Biolux 654
Ezonics EZCam USB II (uvt8532)
Ezonics EZCam USB III
TerraCam USB
Stick WebCam
Mini WebCam
Tucan PenCam
Che-ez! Webbie
SNAKE EYE SI-8480/8481
PC CAM CP03
WEB Camera PBC0006
ClipCam

Ovladače založené na čipové sadě NW802

Tuto čipovou sadu od společnosti DIVIO podporuje ovladač, který naleznete na webu *http://nw802.sourceforge.net.* K podporovaným modelům patří následující:

BTC SurfCam CMOS300k
Mustek WCam 300
Logitech QuickCam Pro USB (starší model „dark focus ring“)

Webkamery Philips na USB

Protože vypršela smlouva „Non-Disclosure-Agreement“ mezi společností Philips Corporation a někdejším správcem ovladače pwc, byla odebrána podpora jádra pro webkamery založené na čipu Philips PWC. Naštěstí se vyvíjí nový ovladač, který nevyžaduje modul, jenž je ve vlastnictví společnosti. Informace a diskusi o změně naleznete na starším webu *http://www.smcc.demon.nl/ webcam/*; nový ovladač je k dispozici na internetové adrese *http://wwwsaillard.org/linux/pwc/*.

K podporovaným modelům Philips patří následující:

PCA645VC

PCA646VC
PCVC675K Vesta, Vesta Pro a Vesta Scan
PCVC720K/40 ToUCam XS, ToUCam Fun, ToUCam Pro a ToUCam Scan
Askey VC010
Creative Labs Webcam 5, Pro Ex
Logitech 3000 and 4000 Pro, Notebook Pro a Zoom
Samsung MPC-C10 a MPC-C30
Sotec Afina Eye
Visionite VCS UM100 a UC300

Ovladač pro kamery založené na čipové sadě SPCA50X

Informace týkající se této čipové sady naleznete na internetové adrese *http://spca50x. sourceforge.net/spca50x.php.* Tento ovladač se neustále vyvíjí a nabízí částečnou nebo úplnou podporu pro následující modely:

Kodak DVC-325 a EZ200
Creative PC-CAM 300, 600, 750
Genius VideoCAM Express V2
Micro Innovation IC 200/IC 150
Logitech ClickSmart 310, 420, 510, 820 a modely Cordless
Logitech Pocket750
Benq DC 1016, 1300, 1500, 3410
Flexcam 100
Aiptek MegaCam, [1.3 Megapixel] Mini PenCam a PocketCam 1.3M Smart
Finet Technology Palmpix DC-85
Pure DigitalDakota
3Com Home Connect lite
Megapix V4
Mustek gSmart: Mini, Mini2, Mini3, LCD 2, LCD 3
Digital Dream Enigma 1.3, Epsilon 1.3
Maxwell Compact Pc PM3
Modely Jenoptik
Minton S-Cam F5
D-Link DSC-350
Trust FamilyC@m 300 Movie
Aiptek Pocket DV, PocketDVII, DV3100+, mini PenCam 2, PocketCam 3M, Pencam SD 2, Pocket DV3500
Hama Sightcam 100
Micro Innovations IC50C, IC400c
FlyCam USB100
Arowana USB Camera 300 K
Intel Easy PC Camera, CS120 (Easy PC Share), PC Camera Pro (CS431), Pocket PC Camera (CS630)
Grandtec V.cap
Sigma-Apo Petcam

Modely založené na čipové sadě STV0680

Webkameru USB vyrobenou s touto čipovou sadou podporují verze jádra 2.4.18 a vyšší s modu-lem stv680.o. Na internetové adrese *http://stv0680-usb.sourceforge.net/* můžete také získat zdroj. Tento ovladač podporuje modely, jako jsou Aiptek Pencam a Nisis Quickpix 2.

Používáte-li sériovou verzi, z nichž hlavní je Scan e-Studio, měli byste navštívit web *http://stv0680.sourceforge.net/*.

Winbond w9966cf

Ovladač pro rozhraní s paralelním portem, který podporuje řídicí čip Philips SAA7111 CCD, s kte-rým se setkáte u webkamery Lifeview Flycam SUPRA. Je součástí jádra verzí 2.4 a novějších pod hlavičkou podpory „video4linux“. Domovskou stránku tohoto projektu naleznete na internetové adrese *http://hem.fyristorg.com/mogul/w9966.html*.

Webkamery založené na čipové sadě Xirlink C-it™ HDCS-1000

Teto ovladač je určen pro webkamery USB vyrobené společnostmi Xirlink, IBM (PC Camera) a Veo Stingray. Podpora je k dispozici v sekci USB jádra Linux od verze 2.2.12. Další informace naleznete na webu *http://www.linux-usb.org/ibmcam*.

Přístup k videozařízení

Následující kapitola platí pro všechny typy připojení.

Videozařízení

Jádro Linux vyžaduje vytvoření „uzlu“ virtuálního zařízení pro přístup a řízení příslušného zaříze-ní. Možná byl již vytvořen při spouštění; existenci videozařízení můžete zjistit příkazem ls -l /dev/video* (s hvězdičkou), případně find /dev -name video* nebo i vizuální prohlídkou adre-sáře /dev. Pokud tomu tak je, můžete přejít ke kapitole „Skupiny a oprávnění“; v opačném pří-padě je budete muset vytvořit ručně.

Snadným způsobem jejich vytvoření je použití skriptu MAKEDEV, který může být umístěn v adre-sáři /dev nebo na obvyklých místech určených k ukládání spustitelných příkazů (/bin, /sbin atd.). Dále vás může nasměrovat manuálová stránka pro MAKEDEV (man MAKEDEV), nezapomí-nejte však na možnosti příkazů týkající se konkrétního zařízení. Pokud MAKEDEV nefunguje či nee-xistuje, přejděte k dalšímu odstavci.
Zařízení můžete vytvořit jako blok (například jako jednotku), jako fifo (file-in-file-out nebo rouru, jako v xconsole) nebo znakové zařízení, které zastupuje jiný hardware. Každé zařízení má hlavní a vedlejší číslo „souřadnice“, kterým jádru oznamuje, čím je a jakým způsobem se k němu má při-stupovat. Tato čísla nejsou libovolná. Hlavní číslo 81 s vedlejším číslem 0, 1, 2 atd. je podle kon-vence přiřazeno zařízením Video4linux, včetně tunerů TV a webkamer. Pro vytvoření videozaří-zení /dev/video0 použijte v příkazovém řádku příkaz mknod:

```
# mknod /dev/video0 c 81 0
```

kde c představuje znakové zařízení.

Můžete použít následující skript, který jsme si zapůjčili ze zdroje jádra (jeho umístění je linux/Documentation/video4linux/bttv/MAKEDEV):

```
#!/bin/bash function makedev () { for dev in 0 1 2 3; do echo "/dev/$1$dev: char
81 $[ $2 + $dev ]" rm -f /dev/$1$dev mknod /dev/$1$dev c 81 $[ $2 + $dev ]
chmod 666 /dev/$1$dev done

    # symlink for default device
    rm -f /dev/$1 ln -s /dev/${1}0 /dev/$1
    }


    # see http://roadrunner.swansea.uk.linux.org/v4lapi.shtml echo "*** new device names ***" makedev video 0 makedev radio 64
    makedev vtx 192 makedev vbi 224 # "*** old device names (for compatibility only) ***" #makedev bttv 0 #makedev bttv-fm 64
    #makedev bttv-vbi 224
```

Jednoduše zkopírujte výše uvedený skript do svého oblíbeného editoru, uložte jej jako MAKEDEV nebo pod jakýmkoli jiným názvem, zajistěte jeho spustitelnost (tj. chmod u+x MAKEDEV) a poté jej spusťte jako root:

```
# ./MAKEDEV
```

Skupiny a oprávnění

Po zavedení všech modulů a vytvoření uzlů zařízení byste se měli ujistit, zda váš uživatelský účet může k zařízení přistupovat. Nejbezpečnějším způsobem je přidání přístupu pro určitou skupinu. V mém systému mají členové skupiny „video“ oprávnění používat webkameru, skener a další foto-grafická zařízení. Nejdříve je nutné změnit vlastnictví zařízení v /dev takto (jako root):

```
# chown root.video /dev/usb/video*
```

kde root.video je vlastník a skupina, ke které bude zařízení náležet. Konkrétní příkaz se samo-zřejmě bude lišit podle systému a typu zařízení. Je důležité, abyste změnili vlastnictví uzlu zaříze-ní jako takového a ne symlink; vlastnictví symlinků ovlivňuje pouze změna nadřazených zařízení či souborů, na které odkazují.

Chcete-li zjistit, zda je váš uživatelský účet členem určité skupiny, spusťte jako root následující pří-kaz: grep -e video /etc/group. Měli byste vidět výpis podobný tomuto:

```
video:x:44:
```

...kde „44“ je číslo skupiny. Protože za poslední dvojtečkou ve skupině „video“ nejsou žádní členové, můžeme přidat například uživatele „jhs“ za použití příkazu:

```
# adduser jhs video
```

Poté již jednoduše povolíte přístup pro čtení a zápis dané skupině:

```
# chmod g+rw /dev/v4l/video0
```

kde g+rw znamená přístup pro čtení (r) a zápis (w) pro skupinu (g). Další informace o příkazu chmod (man chmod nebo info chmod) naleznete v dokumentaci.

Aplikace pro snímání

Programy příkazového řádku

Jak již název napovídá, nevyžadují tyto programy systém X Window pro funkce kamery a ukládání snímků.

Streamer

Streamer je mnohostranný program, který umožňuje ukládat snímky z webkamery či videozaříze-ní prostřednictvím příkazového řádku. Může být k dispozici v distribučním balíčku Xawtv, pří-padně jej získáte samostatně, například jako v Debianu. Tento program a další informace nalez-nete na webu Gerda Knorra (*http://www.bytesex.org/xawtv*).

Pro uložení standardního obrázku JPEG z příkazového řádku, v kterém se ke kameře přistupuje pomocí příkazu /dev/video0, postupujte takto:

```
$ streamer -c /dev/video0 -b 16 -o outfile.jpeg
```

kde -b je číslo barev (v bpp buď 15, 16, 24 nebo 32) a -o je výstupní název souboru, který se uloží do aktuálního adresáře (pro jiné umístění určete -o /cesta/vystupnisoubor.jpg). Pokud chce-te uložit více obrázků, nezapomeňte připojit k názvu souboru nuly, aby program mohl uložit obrázky s pořadovými čísly, tj. -o outfile000.jpeg bude outfile001.jpeg, outfile002.jpeg atd.

Pro vytvoření souboru .avi:

```
$ streamer -q -c /dev/video0 -f rgb24 -r 3 -t 00:30:00 -o /home/jhs/outfile.avi
```

...kde -q znamená „tiché“ spuštění (žádná výstupní zpráva), -f je „formát“ (rgb24 je TrueColor avi), -r určuje počet rámců za sekundu a -t je doba záznamu (30 minut). Program Streamer může uklá-dat formáty Quicktime™ (non-Sorensen) a ukládá také audio. Více informací získáte po zadání pří-kazu streamer --help.

camE

CamE je program příkazového řádku, který pracuje jako démon a ukládá rámce z vašeho zaříze-ní v4l pro archivaci či načítání (například na webserver) prostřednictvím ftp či scp. Úpravou pří-slušného řádku v konfiguračním souboru můžete překrývat další grafické prvky, časovat rámce nebo přidávat dynamický text. Další informace je možno nalézt na webu camE (*http://linuxbrit.co.uk/camE/*).

Motion

Motion je skvělý program, který je schopný sledovat videosignál z jedné či více webkamer. Může zaznamenávat periodické snímky a při detekci pohybu nahrává ve formátu mpeg a/nebo provádí jinou akci, jako je odeslání e-mailu nebo spuštění příkazu. Může sledovat a graficky označit dete-kovaný pohyb, doplňovat soubory prostřednictvím serveru http na váš web, předávat je do jiné aplikace a provádí i další činnosti. Počet možností příkazového řádku je ohromující; existuje však web *http://www.lavrsen.dk/twiki/bin/view/Motion/MotionGuide*, který velmi pěkně popisuje různé možnosti příkazů a konfiguračních souborů. Domovskou stránku programu Motion naleznete na internetové adrese *http://www.lavrsen.dk/twiki/bin/view/Motion/WebHome*.

Webcam

Webcam je automatizovaný nástroj příkazového řádku pro práci s webkamerou, který je k dispo-zici například na webu Xawtv (*http://www.bytesex.org/xawtv*). Je vhodný pro automatizované ope-race, protože nevyžaduje žádné parametry příkazového řádku, pouze předem upravený konfigu-rační soubor (obvykle ~/.webcamrc). Podobá se výše zmiňovanému programu camE v tom, že může snímat obrázky a nahrávat je na webový server prostřednictvím ftp či ssh.

SANE

Program SANE (Scanner Access Now Easy) podporuje přístup zařízení v4l, včetně webkamery v novějších verzích. Pokud jste v systému Linux již pracovali s fotografickým skenerem, mohli byste program SANE používat pro snímání obrázků, zvláště v případě zařízení, která fungují jako skener i jako digitální kamera.

Programy založené na grafickém uživatelském prostředí Xawtv

Xawtv je program pro přístup k videozařízením v systému Linux, včetně tunerů TV a webkamer.

Domovskou stránku naleznete na internetové adrese *http://bytesex.org/xawtv*. Když poprvé vyzkoušíte svou webkameru a konfigurace je podle vás správná, použijte parametr-hwscan:

```
$ xawtv -hwscan This is xawtv-3.72, running on Linux/i686 (2.4.21) looking for available devices

/dev/v4l/video0: OK [ -device /dev/v4l/video0 ] type : v4l name : BT878(Hauppauge (bt878)) flags: overlay capture tuner
/dev/v4l/video1: OK [ -device /dev/v4l/video1 ] type : v4l name : IBM USB Camera flags: capture
```

aby se zobrazila dostupná zařízení (výstup se může lišit). Zkuste otevřít xterm a spustit xawtv, přičemž budete snímat ze své webkamery:

```
$ xawtv -c /dev/video1 This is xawtv-3.72, running on Linux/i686 (2.4.21)
```

A (doufejme) vaše kamera začne snímat obrázky do okna na monitoru. V prostředí xterm se mohou zobrazit chybové zprávy, které vám mohou pomoci při zjišťování potíží s konfigurací. Pokud vás tyto informace nezajímají a vše funguje dobře, můžete při příštím spuštění programu použít nabídku správce oken. Více možností programu xawtv získáte pomocí příkazu man xawtv.

Gqcam

Gqcam (*http://cse.unl.edu/~cluening/gqcam/*) je grafická aplikace založená na GTK+, která byla původně vytvořena pro přístup ke kamerám Connectix QuickCams, nyní však podporuje téměř všechny webkamery kompatibilní s Video4Linux. Má intuitivní rozhraní, které usnadňuje zobra-zování, snímání a konfiguraci nastavení webkamery. Je vhodná pro ty, kteří chtějí pouze snímat obrázky a nechystají se upravovat konfigurační soubor nebo používat příkazový řádek.

Camorama

Camorama je grafická aplikace založená na GTK+2.0, která se velmi podobá programu gqcam vytvořenému pro Gnome2. Domovskou stránku této aplikace naleznete na internetové adrese
http://camorama.fixedgear.org/.

Ekig (GnomeMeeting)

Ekiga (původně GnomeMeeting) je aplikace VOIP/IP-Telephony pro desktop Gnome2, která podporuje i videokonference prostřednictvím webkamer. Podrobné informace naleznete na webu
http://www.gnomemeeting.org/.

Řešení potíží

Mám webkameru USB a nevím, o jaký model se jedná a/nebo kdo je výrobce. Co mám dělat?

Použijte příkaz lsusb; díky tomuto příkazu můžete zjistit, jaká další zařízení USB jsou dostupná ve vašem systému:

```
$ lsusb
Bus 007 Device 001: ID 0000:0000
Bus 006 Device 001: ID 0000:0000
Bus 005 Device 001: ID 0000:0000
Bus 004 Device 001: ID 0000:0000
Bus 003 Device 003: ID 0545:8080 Xirlink, Inc. IBM C-It WebCam
Bus 003 Device 002: ID 046d:0840 Logitech, Inc. QuickCam Express
Bus 003 Device 001: ID 0000:0000
Bus 002 Device 003: ID 051d:0002 American Power Conversion Back-UPS Pro 500/1000/1500
Bus 002 Device 001: ID 0000:0000
Bus 001 Device 001: ID 0000:0000
```

Čísla uvedená za „ID“ zastupují výrobce a produkt. Vyhledat je můžete v katalogu Linux USB ID (*http://www.linux-usb.org/usb.ids*).Nemůžete-li lsusb použít a máte podporu pro systém souborů /proc a podporu pro systém souborů USB, zadejte v příkazovém řádku následující:

```
$ cat /proc/bus/usb/devices
```

Měli byste získat výstup obsahující toto:

```
T: Bus=01 Lev=01 Prnt=01 Port=01 Cnt=01 Dev#=3 Spd=12 MxCh=0
D: Ver= 1.01 Cls=ff(vend.) Sub=ff Prot=ff MxPS=8 #Cfgs=1
P: Vendor=0545 ProdID=8080 Rev= 3.0a
S: Product=USB IMAGING DEVICE
```

Řádek začínající na „T:“ je sběrnice USB, ke které je zařízení připojeno. „P:“ označuje identifikátor výrobce a produktu, který je uveden v seznamu na webu *http://www.linux-usb.org*.

Nemohu najít kameru v /dev!

Pokud předpokládáte, že je typ připojení podporován a kamera funguje, nahlédněte do kapitoly „Uzel videozařízení“.

Vidím zařízení kamery (fyzicky i jako uzel zařízení v /dev), ale nemám k němu

přístup!

Viz kapitola „Skupiny a oprávnění“.

Moje kamera má ovladač, který je pouze zdrojový, tj. musím jej vytvořit! Kde mám začít?

Nejdříve zjistěte, zda vaše distribuce systému Linux nabízí předkompilovaný binární soubor ovla-dače. Ten pak můžete načíst stejným způsobem, jakým byste postupovali u modulu. V opačném případě se ujistěte, že jsou nainstalované zdroje jádra. V závislosti na distribučním balíčku bude-te potřebovat také GNU, gcc, binutils a možná i další programy. (Uživatelé Debian by si měli pře-číst následující část, v které naleznou instrukce týkající se daného distribučního balíčku.)

Stáhněte si zdroj ovladače (v tomto příkladu s názvem src.tar.gz) a rozbalte jej:

```
$ tar -xvzf src.tar.gz
```

Poté přejděte na adresář se zdrojem jádra:

```
# cd /usr/src/linux
```

Vytvořte potřebné zdrojové soubory:

```
# make oldconfig# make dep
```

Nyní přejděte na adresář, do kterého jste rozbalili zdroj ovladače, a v souborech README a/nebo INSTALL si přečtěte pokyny k vytvoření ovladače. Obvykle se jedná o určitou kombinaci příkazů make, make all a/nebo make install. Za předpokladu, že kompilace proběhla správně, můžete nový modul jednoduše načíst pomocí příkazu modprobe. Dojde-li k nějakým potížím, nahléd-něte do kapitoly „Kde najdu další informace?“.

Používám Debian GNU/Linux. Existuje jednodušší způsob kompilace jádra a vytvoření zdrojových modulů?

Je mnohem jednodušší použít automatizovaný nástroj kernel-package. Nejdříve jej nainstalujte pomocí příkazu apt-get. Poté nainstalujte požadovaný zdroj jádra (například apt-get install ker-nel-source-2.X.X). Rozbalte zdroj jádra (ve formátu .bz2) prostřednictvím tar -xvjf a poté vytvoř-te symbolický odkaz s názvem linux, který odkazuje na nový zdroj:

```
# ln -s /usr/src/kernel-source-2.X.X /usr/src/linux
```

Poté použijte cd /usr/src/linux a příkaz: make-kpkg clean, za kterým následuje make menu-config či make xconfig, jako při kompilaci nového jádra. Potom můžete použít make-kpkg ker-nel_image a nainstalovat nový balíček jádra, který jste uschovali v /usr/src pomocí dpkg -i ../kernel-image-2.X.X. Následně můžete prostřednictvím apt získat balíček zdrojového ovla-dače. V případě Quickcam Express je balíček qc-usb-source:

```
# apt-get install qc-usb-source
```

a rozbalte archív:

```
# tar -xvzf qc-usb-modules.tar.gz
```

Zdroj se rozbalí do adresáře /usr/src/modules. Posledním krokem v /usr/src/linux je vytvo-ření modulů s balíčkem jádra:

```
# make-kpkg modules_image
```

Nainstalujte nový balíček do /usr/src/ (s názvem qc-usb-modules-[arch].deb) za použití dpkg -i. Nakonec načtěte modul:

```
# modprobe quickcam
```

Informace o možných potížích naleznete v dokumentaci /usr/share/doc/kernel-...

Moji kameru podporuje ovladač, který se musí nainstalovat jako patch do jádra! Co mám dělat?

Podrobné informace naleznete na internetové adrese *http://tldp.org/HOWTO/Kernel-HOWTO/patching_the_kernel.html* na webu The Linux Documentation Project (*http://tldp.org/*). Stručná a nezaručená verze instalace patche je následující: Ujistěte se, že jsou splněné podmínky popisované v kapitole „Moje kamera má ovladač, který je pouze zdrojový, tj. musím jej vytvořit! Kde mám začít?“. Nejdříve přejděte v příkazovém řádku nebo terminálu xterm do zdrojového adresáře verze jádra, do něhož budete instalovat patch kamery (v tomto příkladu se jedná o patch.diff).

```
# cd /usr/src/linux # patch -p1 -E patch.diff
```

Mělo by se zobrazit potvrzení o úspěšné instalaci patche. Nyní můžete použít příkaz make menu-config nebo jakýkoli jiný

program pro překompilování, čímž povolíte příslušnou podporu. Pokud se instalace nezdaří nebo dojde k jiným potížím, měli byste zobrazit dokumentaci man patch a přečíst si kapitolu „Kde najdu další informace?“.

Kde najdu další informace?

Navštivte web *https://listman.redhat.com/mailman/listinfo/video4linux-list.*

Chci přispět k podpoře Video4Linux v systému Linux! Koho mohu kontaktovat?

Navštivte web *http://www.thedirks.org/v4l2/peopleprojects.htm.*

Quake

Úvod

Předmluva

Hra Quake v prostředí operačního systému Linux spojuje dvě revoluční záležitosti počítačového světa – populární, volně dostupný operační systém GNU/Linux a GLQuake, první „střílečku“, která využívala trojrozměrnou grafickou knihovnu OpenGL. Hra Quake spojuje věrnou komunitu lidí tvořící hry i deset let poté, co ji společnost ID Software uvedla, a díky Johnu Carmackovi, který software uvedl pod GPL, si lidé mohou hru užívat i na otevřených platformách. Znamená to vše, že je tato hra tak důležitá? Ne, samozřejmě jde hlavně o zábavu.

Dokument

Tento dokument není vyčerpávajícím návodem pro nastavení jakéhokoli programu Quake, ale je moderním průvodcem hrou GLQuake, souhrnem nejužitečnějších tipů a upozorňuje na nejlepší zdroje dostupné na webu. Zaměřuje se na četná přepracování modulu a skvělé doplňky, které z hry Quake činí fenomén.

Obecné informace

Začínáme

Instalace hry Quake vyžaduje několik základních kroků.

Získání datových souborů hry

Datové soubory hry získáte instalací hry v prostředí operačního systému Microsoft Windows nebo prostřednictvím některého z emulátorů (jako je například Wine nebo Dosbox) a následného zko-pírování adresáře id1 instalované hry do adresáře Linux Quake (ujistěte se, že jste všechny názvy souborů zadali malými písmeny).

Pokud vlastníte CD DOS Quake, můžete datové soubory získat pomocí nástroje lha (*http://freshmeat.net/projects/lhaforunix/*). V případě starších CD musíte nejprve zadat

```
cat /mnt/cdrom/quake101.1 /mnt/cdrom/quake101.2 > resource.1
```

Jakmile máte soubor resource.1, zadejte cd /usr/local/games/quake lha e
{some directory}/resource.1
Novější verze hry, včetně WinQuake, obsahují datové soubory v nezkomprimovaném tvaru a adre-sář id1 lze zkopírovat přímo z CD.

Instalace spustitelných programů Quake

Program pro spouštění hry Quake se označuje jako herní 'engine'. Můžete si vybrat z několika programů, nejjednodušší volbou je GLQuake nebo upravený TyrQuake (*http://prdownloads. sourceforge.net/uhexen2/tyr-glquake?download*). Poté co stáhnete nebo zkompilujete některý z těchto programů, umístěte tento binární soubor do svého adresáře Quake spolu s adresářem id1.

Shrnutí

Struktura adresářů by měla vypadat podobně jako na níže uvedeném schématu. Názvy adresářů je nutné zadat malými písmeny:

```
/usr/local/games/quake +- glquake.glx (herní engine)
                       |
                       +- id1 -+- config.cfg
                               |
                               +- game.dat
                               |
                               +- pak0.pak
                               |
                               +- pak1.pak
```

Pro spuštění hry zadejte v okně xterm následující:

```
./glquake.glx -fullscreen -width 800 -height 600
```

Nápověda

Tento postup možná vypadá jednoduše, ale pokud je pro vás prostředí systému Linux nové a nej-ste zvyklí pracovat s příkazovým řádkem, mohou být pro vás užitečné informace v následujících dokumentech nebo webech:

Část Řešení potíží
Web Linuxgamers Quake (*http://www.linux-gamers.net/modules/wfsection/article.php? articleid=42*)
Web Quake wiki (*http://wiki.quakesrc.org/index.php/HomePage*)
Web The Linux Game Tome (*http://www.happypenguin.org/*)

Možnosti příkazového řádku

Možnosti příkazového řádku jsou zvláštní parametry uvedené za názvem programu v příkazovém řádku systému Linux. Quake takových možností nabízí mnoho a některé se u jednotlivých modu-lů liší. Níže uvádíme některé z užitečných a často používaných možností:

-window

Spustí hru v režimu okna.

-fullscreen

Spustí hru na celé obrazovce.

-game NAME

Nahraje mód (mod) jménem NAME.

-mem N

Rezervuje určitou velikost pro vnitřní paměť. Většinou stačí 8 či 16 megabajtů, ale měli byste ji zvýšit, pokud hrajete v rozsáhlejších módech.

-width WIDTH

Šířka okna/celá obrazovka.

-height HEIGHT

Výška okna/celá obrazovka.

-sndspeed MHz

Nastaví hodnotu frekvence zvuku (například 44100, 22100, 11025).

-sndbits N

Nastaví počet zvukových bitů tak, že N = 8 nebo 16.

-nosound

Vypnutí zvuku. Toto nastavení je nutné, pokud zvuk není nakonfigurován, abyste předešli „zamrzání“ hry.

-nomtex

Zakáže zobrazování vícenásobné textury GL.

-listen N

Povolí maximální počet hráčů, kteří se mohou připojit ke hře pro více hráčů.

-cddev DEVICE

Použije DEVICE k přehrání disku CD.

-nocdaudio

Zakáže zvukové CD.

Příkazy hry Quake lze do příkazového řádku v prostředí Linux přidat pomocí znaménka plus. Například pro automatické spuštění hry s vysokou obtížností zadáte tento příkaz:

glquake.glx +skill 2 +map e1m1

Příkazy hrací konzoly

Více informací naleznete na internetové adrese http://www.planetquake.com/console/ commands/quake.html.
Konzola je příkazový řádek „uvnitř“ hry, jehož prostřednictvím můžete zadávat příkazy, měnit pod-mínky a používat cheaty. Objeví se po stisknutí klávesy ~ ve hře. K hlavním příkazům patří: god

Nezranitelnost.

noclip

Procházení zdmi.

notarget

Nepřátelé neútočí na hráče.

timedemo DEMO

Přehrávání DEMO (například „demo1“) na nejvyšší rychlost a zobrazení frekvence snímků.

impulse N

Vydá „počet impulzů N“. Díky těmto příkazům ve hře získáte speciální funkce. Nejčastěji se používá příkaz impulse 9, jehož prostřednictvím získáte všechny dostupné zbraně. bind key „COMMAND“ Nastaví na klávesu key příkaz COMMAND.

map MAP

Nahraje mapu.

changelevel MAP

Nahraje mapu, aniž by smazal hráčovo nastavení.

quit

Návrat do systému.

skill VALUE

value = 0 (jednoduché) – 3 (nemožné)

Stupeň obtížnosti. Úroveň je nutné restartovat.

r_wateralpha VALUE

value = 0.0–1.0

Neprůsvitnost vody.

_snd_mixahead VALUE

value = 0.1–1.0 Zvýšení této hodnoty je dobrým způsobem zrychlení hry, za cenu prodlevy zvuku. Přiměřená je hodnota 0.3.

r_shadows FLAG

flag = 0 | 1

Zobrazí stíny.

vid_wait FLAG

flag = 0 | 1

Synchronizuje výstup videa s obnovováním obrazovky.

chase_active FLAG

flag = 0 | 1

Zobrazuje hráče z pohledu třetí osoby.

Herní engine

GLQuake

Na webu *http://mfcn.ilo.de/glxquake* naleznete související dokumentaci, tipy pro odstranění pro-blémů a také základní verzi OpenGL Quake pro prostředí systému Linux. Podporuje většinu módů Quake, nefunguje však podpora gamma (jas).

Darkplaces

Darkplaces je skvělý engine hry Quake se širokou škálou vizuálních efektů a možnostmi barev,efektů a zvuku. Používá stejné osvětlovací efekty jako Tenebrae, a vyžaduje proto výkonnější počí-tač než GLQuake a QuakeForge.

Podporuje také mnoho jinak nekompatibilních módů, včetně Nehahra a Nexuiz, přičemž se také

zdokonalila podpora oficiálních balíčků s různými misemi.Havocův archív souborů (*http://icculus.org/twilight/darkplaces/files/*) může být lehce nepřehledný.Nabízí zkompilované binární soubory a zdrojový kód hry. Pro kompilaci svého vlastního progra-mu rozbalte druhý archiv a zadejte příkaz make. Zobrazí se seznam možných voleb pro kompi-laci, z nichž jednu vyberte. Například k vytvoření enginu OpenGL se zvukem typu ALSA zadejtemake cl-release, pro typ zvuku OSS zadejte příkaz make cl-release DP_SOUND_API=OSS.

Porty QuDos Quake

Na webu *http://qudos.quakedev.com/linux/quake1* naleznete enginy (včetně DemonQuake, Joe-Quake, NehQuake, Qrack a Tremor), které byly poprvé portovány pro prostředí Linuxu. NehQu-ake (*http://qudos.quakedev.com/linux/quake1/NehQuake/-bjp-bin-src.linux.tar.bz2*) spustí mód Nehahra mnohem rychleji než engine Darkplaces.

Archivy obsahují zdrojový kód, binární soubory a ve většině případů i datové soubory nezbytné pro funkci enginu. Pro instalaci těchto datových souborů vyhledejte jednoduše pojmenované slož-ky (jako například joequake nebo qrack) a přesuňte je do svého adresáře quake. Pokud je nena-jdete, budete je muset stáhnout z domovské stránky hry.

Pro některé enginy je nutné nainstalovat přídavnou knihovnu zvuku (která je jejich součástí). Postupujte takto:

su cd /usr/lib mv {some directory}/libfmod-3.74.1.so . ln -s libfmod-3.74.1.so libfmod.so ldconfig

nebo navštivte web FMOD (*http://www.fmod.org/*) a nainstalujte knihovny ručně.

QuakeForge

QuakeForge je nejobsáhlejší projekt Quake pro prostředí Linux. Nabízí opticky zdokonalený engine, mnoho klientů pro jednoho hráče a QuakeWorld a nástroje Quake C. Mezi jeho vlastnosti patří přehledný systém menu, nový „systém upozornění“ a nápověda uvnitř hry.

Zřejmě z důvodu své velikosti nebyl projekt QuakeForge po léta aktualizován a jeho dokumenta-ce nebyla nikdy úplně dokončena. Pomocí obvyklého „configure && make && make install“ může být zkompilován celý projekt, ale pokud chcete vytvořit pouze binární soubor pro jednoho hráče, zkuste následující:

configure --with-static-plugins --without-tools --without-servers --with-clients=glx zcat <ruamoko/cl_menu/menu.dat.gz >{somedir}/quake/id1/menu.dat

Více informací o vytvoření QuakeForge v prostředí BSD Unices naleznete v části FreeBSD. Infor-mace o QuakeForge jsou k dispozici na internetových adresách *http://www.quakeforge.net* a *http://sourceforge.net/projects/quake/*.

TyrQuake

TyrQuake (*http://disenchant.net/engine.html*) je obsáhlý projekt, který se věnuje hře Quake, kli-entům QuakeWorld a dalším nástrojům (*http://disenchant.net/utils.html*), včetně oblíbeného nástroje TyrLite. Tyrann se zaměřuje na plně vybavené a přitom minimalistické enginy pro ope-rační systémy Windows a Linux.

Pro kompilaci TyrQuake-0.47 nejprve upravte soubor Makefile a vyberte některé možnosti. (Kli-ent pro jednoho hráče je „NQ“). Uživatelé grafických karet Nvidia mohou k opravě několika chyb použít patch, který je k dispozici na webu *http://prdownloads.sourceforge.net/uhexen2/tyrquake-0.47-nvidia.patch?download.* U novějších verzí (po 0.47) by neměla být instalace tohoto oprav-ného balíčku nutná.

Opravený binární soubor pro jednoho hráče TyrQuake naleznete na internetové adrese *http://prdownloads.sourceforge.net/uhexen2/tyr-tlquake?download.*

Quake Software

Zdroj WinQuake (*http://www.quakeforge.net/files/q1source.zip*) byl vytvořen ve dvou verzích s různým rozlišením:

X Quake (quake.x11)
Svga Quake (squake)

Jejich kompilace však již není jednoduchá. Je třeba zkopírovat soubor Makefile.linux do Make-file, upravit tento soubor odstraněním přebytečných úkolů, nahradit /usr/X11/lib pomocí /usr/X11R6/lib a zadat příkaz make build_release.

Existují ale i jednodušší možnosti. TyrQuake a QuakeForge obsahují softwarové klienty a je zde také starší SDL Quake (*http://www.libsdl.org/projects/quake/*) vytvořený autorem SDL Samem Lan-tingou, který by měl fungovat na všech moderních platformách.

Jiné enginy NPRQuake

Dalším enginem Quake je NPRQuake (*http://www.cs.wisc.edu/graphics/Gallery/NPRQuake/*), který byl portován pro prostředí Linux, ale několik let nebyl s největší pravděpodobností nijak upravo-ván. Především nabízí možnost nahrát různé metody vykreslování za běhu(!). Linux port (*http://www.geocities.com/coolguywithgun*) obsahuje vykreslování pomocí ainpr (*http://www.cs.unc.edu/~adyilie/comp238/Final/Final.htm*) a osobně jsem s ním velmi spokojen.

Verze SDL (*http://www.tempestgames.com/ryan/*) obsahuje přepracovaný kód pro myš a video a díky tomu by měla fungovat i na jinak problémových systémech. Do verze SDL však nebyla importována rozhraní API, takže ve skutečnosti nejde o přenosný engine.

Tenebrae

Tenebrae (*http://tenebrae.sourceforge.net/*) je engine hry Quake s moderním světelným modelem podobným tomu, s jakým se setkáte například ve hře Doom III. Vyžaduje velmi rychlý počítač a nemusí být kompatibilní s veškerým hardwarem.

Projekt Twilight

Projekt Twilight (*http://icculus.org/twilight*) je sadou spíše minimalistických enginů NQ a QW, která se zaměřuje na vysokou rychlost zobrazování, ale v současné době je bohužel trochu nesta-bilní.

Módy

Módy

V síti Internet naleznete stovky úrovní vytvořených uživateli – známé pod názvy „módy“, „předě-lávky“ nebo prostě „mapy“. Při hledání zjistíte, že spousta adres URL už neplatí a mnohem jed-nodušší je zadat ve vyhledávači přímo název souboru (například slovo „quake”), nežli pátrat po domovské stránce projektu.

Pro spuštění nových map umístěte soubor bsp do podadresáře quake/id1/maps a spusťte Quake
za použití příkazu +map MAPNAME.Pro instalaci módů stačí vytvořit podadresář DIR a extrahovat do něj archiv zip/tar. Ten se potomnahraje pomocí možnosti příkazového řádku -game DIR, někdy také +map MAPNAME, kdeMAPNAME je počáteční mapa módu.
Mapy i módy byste měli zadávat malými písmeny.Web QuakeTerminus obsahuje dobrý seznam módů (*http://quaketerminus.com/addon.htm*) a webTenfour mnoho přehledů map (*http://tenfourmaps.telefragged.com/php/revidx.php?gameid=q1&sort-by=date&reversesort=1&page=1*).

Soul of Evil

> Soul of Evil (*http://www.planetquake.com/tronyn/soul*) je skvělým módem se středověkou téma-tikou. Obsahuje dvě epizody pro jednoho hráče, arénu pro bitky a pěknou dokumentaci.

Nehahra

> Výpravný, obtížný a tajuplný mód. Nehahra (*http://www.planetquake.com/nehahra*) je plno-hodnotným módem Quake, v prostředí Linuxu jej podporuje modul Darkplaces, LordHavoc a portem modulu NehQuake. Obsahuje skvělé modely a mapy, dobře využívá efekty mlhy a obsahuje dvouhodinový video/demo (nepovinné), které vás upoutá originálním příběhem hry Quake, je však poněkud zdlouhavé.

Contract Revoked

> Moderní mód (*http://kell.spawnpoint.org/convoked.html*), který vás skutečně upoutá. Obsahu-je i náročné dodatky The Lost Chapters a Quoth.

The Masque of the Red Death

> Nedávno zveřejněná obsáhlá mapa hradu (*http://www.fileplanet.com/dl.aspx?/planetquake/ tronyn/maps/masque.zip*), která vyžaduje modul Darkplaces a rychlý hardware.

The Hunted Chronicles

> Hra The Hunted Chronicles (*http://www.ru1337.com*) vyžaduje engine Darkplaces Hunted Chronicles. Jedná se o dvoudílnou zombie střílečku. Druhá část využívá světelných a mlžných efektů a klidně byste si ji mohli splést s módem Half-Life.

Módy Neila Mankeho

Neil vytvořil plnohodnotnou verzi Half-Life nazvanou They Hunger (*http://www. planethalflife.com/ manke/*). Zábavné a kvalitní jsou i tyto – alba01.zip, alba02.zip, sofsp1.zip, sofsp2.zip, starshp2.zip.

Blood Mage

Blood Mage (*http://www.google.com.au/search?q=bmfull.zip*) je hra plná skvělých monster a hudby. Je trochu zastaralá a obrovské množství kouzel může být únavné.

Operation: Urth Majik

OUM (*http://disenchant.net/files/maps/oum.zip*, *http://www.planetquake.com/fatty/oum/*) je jedním z prvních módů s nádechem sci-fi. Má pět dobře zpracovaných úrovní s mnoha nový-mi zbraněmi.

The Coagula Contest 2

Coagula (*http://www.planetquake.com/underworld/quakerev030814.html*) je sada map se šesti úrovněmi pocházející od konkurence. Novinkou je, že všechny mapy se vznáší v prostoru a jsou skvělé.

Scourge done Slick

SdS (*http://www.planetquake.com/QdQ/sds.html*) je zrychlená verze Mission Pack 1, Scourge of Armagon. Hra je zábavná a jednoduše skvělá.

Insomnia

Skvělá hra plná kaluží krve (*http://www.google.com.au/search?q=czg07.zip*).

Zerstörer

Temná a krvavá klasika s paranoidní atmosférou, kterou známe ze hry Doom. Zerstörer obsa-huje několik dobrých úrovní soubojů (*http://www.google.com.au/search?q=zerstorer.zip*).

Fantasy Quake

FQ (*http://www.google.com.au/search?q=fantasy.zip*) je hra se středověkou tématikou. Má několik skvělých úrovních, ale také mnoho horších částí, protože ve skutečnosti nebyla nikdy dokončena. (Projekt byl předčasně ukončen, když došlo k potížím s hostingem serveru.) Ne všechny třídy hry fungují správně. Dvojím stisknutím klávesy g získáváte předměty, klávesami <, > a / ovládáte inventář a pomocí kláves w, e a r můžete bojovat.

Příručku ke hře Fantasy Quake naleznete na webu *http://prdownloads.sourceforge.net/ uhexen2/fq-manual.tgz?download.*

Gib Factory, Vigil, Museum

Na následujících internetových adresách naleznete malé, ale skvělé módy:

http://www.google.com.au/search?q=gibfact.zip
http://www.google.com.au/search?q=vigil.zip
http://www.google.com.au/search?q=museum.zip

Komerční módy

Mission Pack 1

Scourge of Armagon od Ritual Entertainment (dříve známé pod názvem Hipnotic Interactive).

Mission Pack 2

Dissolution of Eternity od Rogue Entertainment. Oba balíčky misí jsou obecně považovány za lepší, než je původní hra.

Malice

Velmi originální předělávka hry Quake s výborným kulometem. Určitě stojí za to si tento mód koupit.

Abyss of Pandemonium

Komerční mód, nyní zdarma k dostání na webu *http://www.planetquake.com/impel.*

Ravages of Apocalypse

Mód Xmen! Nabízí skvělé modely, ale ovládání má značné nedostatky. Tento mód je nyní zdar-ma k dispozici na webu *http://www.zerogravity.com.au/xmen/.*

Shrak

Jeden z prvních komerčních módů. Skvěle vytvořené a originální příšery, ale hra již není příliš zajímavá.

Nástroje pro tvorbu map

Ve hře Quake byl vytvořen vlastní jazyk – Quake C. Díky tomu mohly módy běžet bez problémů na jakémkoliv operačním systému. Je možné instalovat více editorů, s jejichž pomocí lze vytvářet mapy.

GtkRadiant (*http://www.qeradiant.com/*) je v současné době jediný spravovaný editor na světě.
Quake Army Knife (QuArK – *http://dynamic.gamespy.com/~quark/*) je víceúčelový nástroj, kterýmůže, ale nemusí být podporován systémem Linux.Na webu Func_Msgboard (*http://www.celephais.net/board/forum.php*) můžete sledovat vývoj Q1 a nových projektů mapování.QuakeForge obsahuje nástroje Quake C společně s jejich archivem.Web Quake Wiki (*http://wiki.quakesrc.org/index.php/HomePage*) nabízí několik odkazů souvisejících s operačním systémem Windows.

Pro více hráčů

QuakeWorld

Je vylepšeným enginem pro hru více hráčů po síti nebo on-line. Je začleněn do QuakeForge, FuhQuake a TyrQuake.

FuhQuake

Obsahuje mnoho rozšíření ovládání oproti původní hře QuakeWorld. FuhQuake (*http://www. fuhquake.net/*) se zaměřuje na hru více hráčů a módy pro jednoho hráče nebudou fungovat.

Nexuiz

Nexuiz (*http://www.nexuiz.com*) je nezávislá hra pro více hráčů používající engine Darkplaces. Využívá moderní světelné efekty, takže hráči s méně kvalitními grafickými kartami by se měli pokusit tyto efekty vypnout, aby hra běžela správně.

Digital Paint 2

Paintball přichází do systému Linux! Digital Paint 2 (*http://www.planetquake.com/digitalpaint/*) je hra určená pro více hráčů. Je zalo-žena na enginu Quake II. Hra je pestrá a její zábavná poloha je příjemnou změnou oproti větši-nou temně naladěným hrám Quake. Tato hra je zcela přepracovaná, takže nemusíte vlastnit hru Quake II. Hrát můžete také proti počítači.

Battle Mech

Battle Mech (*http://static.condemned.com/index.shtml*) je skvělý mód ve stylu Mechwarrior. Nutné je použít archiv s verzí 1.1 (*http://static.condemned.com/files/bmech_stuff/battlemech-1.1.tar.gz*)

Řešení potíží

Další možnosti odstraňování chyb naleznete na webu *http://mfcn.ilo.de/glxquake* a na webu *http://www.icculus.org/lgfaq.*

V případě, že používáte alternativní hrací enginy, jako je například TyrQuake, FuhQuake nebo Darkplaces, mohou nastat potíže s myší či zvukem.

Bash nespustí program

"bash: ./glquake.glx: Permission denied"

Binární soubor nemusí mít nastaven příznak spuštění. Tento problém vyřešíte zadáním pří-kazu chmod +x glquake.glx.
Pokud je program umístěn v oddílu Windows, je možné, že ten byl připojen s parametrem noexec.

Zadejte (jako root): mount -o remount,exec /mnt/windows. "bash: glquake.glx: command not found"

- Bash nemusí mít ve své cestě uveden aktuální adresář. Napište „export PATH=$PATH:“.

Program nelze spustit

Zkuste některou z uvedených možností:
Pomocí -nosound vyzkoušejte, zda problém spočívá ve zvuku.Potíže se zvukem podrobně popisujeme níže.
Pomocí -noudp zjistíte, zda je nakonfigurována síť.
Pomocí -nocdaudio zjistíte, zda nechybí CD-ROM.
Prostřednictvím parametrů příkazového řádku -height, -width a -fullscreen vyberete správný režim obrazovky, například glquake.glx -width 800 -height 600 -fullscreen -nosound.

Názvy souborů nejsou zadány malými písmeny nebo chybí datové soubory.

Linux Quake vyžaduje, aby byla většina názvů souborů zadána malými písmeny. Pokud se obje-ví chyba podobná této „Error: W_LoadWadFile: couldn't load gfx.wad“, znamená to, že hra nemůže najít datové soubory. Důvodem může být to, že jejich názvy nejsou zadány malými pís-meny.

Ujistěte se, že podadresář id1 (ne ID1) obsahuje soubory pak0.pak a pak1.pak.
Nástroj pro zajištění malých písmen si můžete stáhnout na webu *http:///prdownloads. sourceforge.net/uhexen2/lowercase?download* nebo *http://filerenameutils.sourceforge.net.*

„Memory overwrite in Sys_Printf“

- Tato chyba znamená, že byste měli v souboru sys_linux.c upravit proceduru Sys_Printf (poblíž řádku 89) změnou text[1024] na text[4096] a následně znovu provést kompilaci.

Program přestane fungovat při nahrávání úrovní

Mnoho módů vyžaduje více paměti. Pomocí možnosti -mem 48 přidáte 48 MB paměti.
Několik novějších módů nebude fungovat se standardním GLQuake a bude vyžadovat rozšířený hrací engine.
V některých případech může tento problém souviset s problémy se zvukem. Vyzkoušejte některé typy uvedené v části o zvuku.

Potíže se zvukem

Více informací o zvuku v systému Linux naleznete v části o ovladačích. Chyba, jako je například „/dev/dsp: Device or resource busy“, znamená, že některý z programů již používá vaši zvuko-vou kartu chcete-li dosáhnout zvukových efektů ve hře Quake, budete muset daný program zavřít.

Pokud chcete ukončit jeden z těchto oblíbených zvukových daemonů, napište do příka-zového řádku Linux příkaz killall artsd nebo killall esd.
Jestliže chcete spustit hru Quake pomocí zvukového daemonu KDE, zadejte příkaz artsd-sp glquake.glx.

„Herní enginy Quake se ukončily a vidím chybu v mmap!“

- Web Linux Gamers FAQ (*http://www.icculus.org/lgfaq*) doporučuje „Vaše zvuková karta/ovladač tuto funkci nepodporuje. Pokud použijete KDE můžete tento problém vyře-šit pomocí přepínače programu artsdsp“.

Ujistěte se, že program artsd běží – po zadání ps -A | grep artsd zkontrolujte, zda tento příkaz vrací alespoň jeden neprázdný řádek. Poté zadejte příkaz artsdsp -m glquake.glx.
Zkuste použít jiné ovladače zvuku uvedené v části o ovladačích.

Zvuk přeskakuje nebo není příliš kvalitní.

- Vyzkoušejte možnosti -sndspeed nebo -sndbits, případně změňte ovladače zvuku.

Další témata Hra je příliš tmavá

Pokud nelze provést změnu nastavení jasu v nabídce možností, můžete pro zesvětlení celé obra-zovky použít program xgamma.

- Před spuštěním hry zadejte xgamma -gamma VALUE. VALUE je hodnota/číslo větší než
  1. Po skončení hry obnovíte nastavení jasu zadáním xgamma -gamma 1.

U nevýkonného hardwaru nebude tento postup účinný. Uživatelé Voodoo 1/2 získají více informací na internetové adrese *http://sourceforge.net/docman/display_doc.php?docid=28982&group_id=124987.*

Pohled prostřednictvím pohybu myši

Hra mi neumožní se pořádně rozhlédnout.

- Pomocí znaku ~ zobrazte herní konzolu a zadejte příkaz +mlook.

Myš nepracuje tak, jak by měla

Zkuste následující postup:

Spusťte hru v režimu celé obrazovky pomocí možnosti -fullscreen.
V hrací konzole zadejte _windowed_mouse 1.
Používáte-li fluxbox, zkuste jiného manažera oken. Fluxbox má potíže s některými hrami v režimu celé obrazovky.
Zatím jste neuspěli? Vyzkoušejte SDL klienty NPRQuake nebo Darkplaces. Zadáním příkazu export SDL_VIDEO_X11_DGAMOUSE=0 před začátkem hry zakážete myš v režimu dga.

Hru/možnosti se nepodařilo uložit

Pokud spustíte Quake jako běžný uživatel a vyskytnou se některé z těchto potíží, může být důvodem to, že nemáte oprávnění zapisovat do adresářů hry. Možné řešení:

Spusťte hru jako superuživatel – než spustíte hru pomocí glquake.glx, zadejte příkaz su a poté heslo uživatele root.
Změňte práva pro adresář hry. Operační systémy Unix mají zabezpečení proti neautorizo-vaným nebo náhodným změnám souborů. Nejjednodušším řešením v uživatelském prostře-dí s jedním uživatelem je, že se stanete superuživatelem a změníte (jako root) vlastnictví adresáře Quake prostřednictvím příkazu chown -R USERNAME /usr/local/games/quake. Uživatelé by si měli přečíst dokumentaci k příkazům chmod a chown.
V prostředí s více uživateli se doporučuje použití herních enginů Darkplaces nebo Qua-keForge, které správně umístí data jednotlivých uživatelů do jejich domovských adresářů. Quake používá poměrně nepřehledný způsob ukládání a obnovení herních možností, zvláště pokud hrajete doplňky. Možnosti hry je někdy nutné znovu inicializovat i přesto, že povolení k souborům není problémem.

„Popletené polygony“

Některé balíčky misí/módy hry Quake mohou způsobit to, že hráč/příšera bude celá zakrytá čára-mi. Tento problém vyřešíte tím, že odstraníte adresář quake/id1/glquake. Při dalším spuštění hry se tento adresář znovu vytvoří a vše by mělo v pořádku.

Čáry na obrazovce

Běžným problémem karet typu 3dfx jsou čáry na obrazovce.

- V hrací konzole zadejte gl_ztrick 0.

Další grafické anomálie

Některé moduly Quake používají urychlovač OpenGL. Většinou pracuje bez problémů, ale pokud se vyskytnou nějaké potíže, zakažte toto zařízení pomocí příkazu -nomtex. Starší grafické karty mohou občas vykreslit objekty v bílé barvě. Na webu PlanetQuake (*http://www.planetquake.com/ console/commands/quake.html*) naleznete seznam příkazů, s jejichž pomocí můžete vyladit výkon.

Ovladače grafiky

Nastavení hardwarové akcelerace GL v systému Linux bývalo poměrně složitou záležitostí, ale sou
časné distribuční balíčky by toto nastavení měly provádět automaticky. Existují samozřejmě výjimky. Ovladače všech moderních grafických karet Nvidia nemají otevřený zdrojový kód. Z tohoto důvo-du je mnoho distribučních balíčků neobsahuje. Pokud je vaše grafická karta Nvidia pomalá, měli byste navštívit stránku *http://www.nvidia.com* a stáhnout si instalátor pro Linux. Dle našich zku-šeností jsou tyto ovladače dobré, ale ne všechny verze fungují stoprocentně se všemi typy karet. Pokud vaše karta nepracuje dobře, zkuste použít jiný ovladač.

Ačkoli nové verze XFree a Xorg podporují Voodoo 3, 4 a 5, starší 3dfx hardware, jako je napří-klad Voodoo1, Voodoo2 a Rush, však již nemají hardwarovou akceleraci. Aby OpenGL pracoval s těmito kartami, jak má, bude nutné stáhnout, nainstalovat a/nebo zkompilovat softwarové kni-hovny zvané Glide a Mesa. Na internetové adrese *http://sourceforge.net/docman/display_ doc.php?docid=28982&group_id=124987* naleznete podrobný soubor README týkající se starých karet 3dfx.

Další odkazy

http://www.x.org

http://www.linux-gamers.net/modules/wfsection/article.php?articleid=22

http://www.linux-gamers.net/modules/wfsection/article.php?articleid=31

Ovladače zvuku

Open Sound System a ALSA jsou dva nejčastěji používané zvukové systémy v prostředí Linux. Pokud máte nějaké potíže se zvukem a nepomohla vám ani část Řešení potíží, měli byste zvážit změnu ovladače zvuku. Je to však poměrně složitá záležitost a pokoušet by se o to měli pouze zkušení uživatelé.

Aktuální ovladač zjistíte zadáním příkazu lsmod. Zobrazí se seznam zavedených modulů jádra. Zvukové moduly ALSA mají dlouhé názvy začínající na „snd“, zatímco moduly OSS mají kratší názvy. Modul ALSA Sound Blaster Live bude mít například název snd-emu10k1, zatímco modul OSS bude emu10k1. U jádra řady 2.4 a starších verzí býval častější modul OSS, dnes se za stan-dard považuje modul ALSA (jádra verze 2.6 a vyšší).

Informace o modulu ALSA naleznete na webu Alsa Homepage (*http://www.alsa-project.org/*)
a v článku Guide to ALSA (*http://www.linuxjournal.com/node/8234/print*) v časopise Linux Journal. Používáte-li modul ALSA a rádi byste vyzkoušeli modul OSS, budete pravděpodobně muset na nových distribucích překompilovat jádro.

Pokračování hry Quake

Hexen II

Hexen II je barvitá adaptace hry Quake, kterou vytvořil tým Raven Software (*http://ravensoft.com/*). Z hlediska použitého

zdrojového kódu a tématu hry se originální hře podo-bá mnohem více než Quake II. Tak pěkná, a přece tak krutá.

Hlavním portem Hexen II v prostředí Linux je Hammer of Thyrion (*http://uhexen2.sourceforge.net/*), u kterého došlo k podstatnému zdokonalení, co se týče chyb a grafiky OpenGL. Přestože není tak rozšířený jako Quake, demoverze hry obsahuje některé z nejlepších úrovní. K dispozici je na webu Sourceforge (*http://sourceforge.net/projects/uhexen2*).

Quake II

Quake II je pokračováním Quake s tématikou sci-fi.První vydání této hry v prostředí Linux nebyla příliš stabilní, problémy byly hlavně s ovládáním myši,ale v současné době již existuje několik souvisejících projektů, z kterých můžete vybírat. Nejnověj-ším z nich je QuDos' Quake II (*http://qudos.quakedev.com/linux/quake2*), který vychází z IcculusQuake II (*http://www.icculus.org/quake2/*), ale má vylepšený grafický engine. Zajímat by vás mohl také projekt QuakeForge Quake II (*http://www.quakeforge.net/files/quake2forge/quake2-0.3.tar.gz*),který podporuje více operačních systémů.Mód hry pro více hráčů Digital Paint 2 vychází z verze Quake II.

Quake III Arena

Třetí verze hry Quake byla mezníkem her pro více hráčů. Byla to jedna z prvních her, jež dosáh

la komerčního uvedení v prostředí Linux.Firma ID Software (*http://www.idsoftware.com/*) nedávno zveřejnila zdrojový kód a na webu Iccu-lus.org naleznete projekt Open Source Quake III (*http://www.icculus.org/quake3/*).
Podporovaný je i komerční doplněk Quake III Team Arena, a přestože nikdy nezískal dobrou kri

tiku, je skvělou hrou.Atmosférickou předělávku Q3 pro jednoho hráče lze získat na webu The Dark Conjunction(*http://www.planetquake.com/tdc/*).

Další odkazy

Instalace Zerowing (*http://zerowing.idsoftware.com/linux/q3a/INSTALL*) a průvodce častými pro-blémy (*http://zerowing.idsoftware.com/linux/q3a/*) od ID Software.Dokumentace k verzi Quake III na webu Linuxgamer (*http://www.linux-gamers.net/modules/wfsection/article.php?articleid=30*).
Fórum o potížích se zvukem (*http://www.linuxquestions.org/questions/history/260975*) a myší (*http://www.linuxquestions.org/questions/history/225821*) ve hře Quake III. Podrobné informace o hře Quake III naleznete na webu Planet Quake (*http://www.planetquake.com/quake3/q3aguide/*).

Quake IV

Hra Quake IV je impozantní FPS od skvělého týmu Raven Software (*http://ravensoft.com/*). Zákla-dem je modul Doom-III a má vysoké požadavky na hardware. Minimálním požadavkem je pro-cesor 2 GHz a paměť 512 MB RAM. Demoverze Quake IV je k dispozici na internetové adrese ftp://ftp.idsoftware.com/idstuff/quake4/demo/quake4-linux-1.0-demo.x86.run.

Další odkazy

http://zerowing.idsoftware.com/linux/quake4/

http://www.linuxquestions.org/