07/98

Linuxové noviny

Úvodem

Pavel Janík ml., 12. července 1998

V červencovém čísle Linuxových novin si můžete přečíst článek Róberta Dobozyho o tom, jak proměnit svůj server na faxový server a tak umožnit svým uživatelům posílat a přijímat faxy nejen na Linuxu, ale i z Microsoft Windows.

V minulých číslech jsme informovali o databázi Interbase a o tom, že již je možno ji provozovat i na Linuxu. Daniel Prynych nám sdělí své zážitky z přenosu Interbase ze systému Windows NT na Linux.

Pokud vám někde v rohu zavazí pracovní stanice SGI Indy či snad SGI Challenge S, tak neváhejte a přečtěte si článek Jana Pazdziory a zkuste těmto strojům vdechnout život a nainstalujte na ně Red Hat Linux.

Milan Šorm nám v první části svého článku ukáže, jak nainstalovat Caldera Netware Server 4.10 Beta. Samozřejmě zde najdete i pokračování seriálu o RPM a také úspěšnou Yenyovu přednášku o hlasových modemech z Linux Installfestu, který se konal v sobotu 18. července v Brně. ■

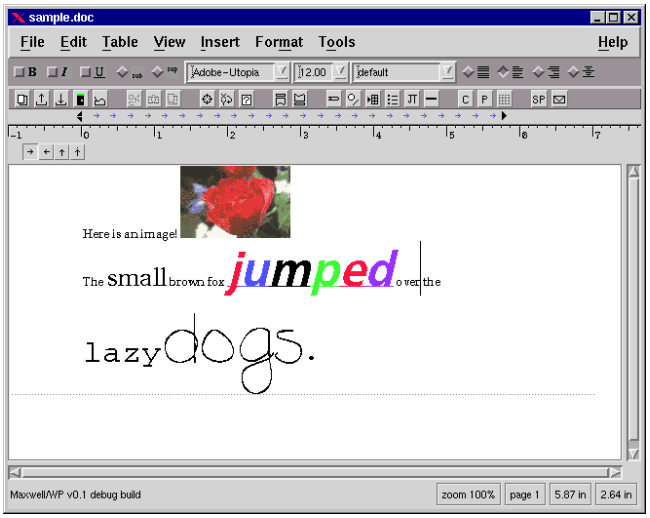
Měsíc v comp.os.linux.announce

Pavel Janík ml., 1. července 1998

Matthias Hoelzer (hoelzer@physik.uni-wuerzburg.de) zveřejnil balík umožňující využívat knihoven Qt a KDE z programovacího jazyka Python. Balík naleznete na adrese (1).

Na adrese (2) najdete seznam 533 diskusních skupin zaměřených na operační systém Linux. Samozřejmostí je vyhledávání, možnost přihlásit se a odhlásit se. Velice zajímavá služba.

Word processor Maxwell byl uvolněn pod licencí GPL. Maxwell by se mohl stát novým standardem v Linuxu — zkuste si jej (3).



Dirk Laessig (dirk@bredex.de) oznámil novou verzi svého programu pro vývoj grafického uživatelského prostředí pomocí knihovny Motif — VDX 1.2. Balík je šířený komerčně, evaluation verzi naleznete na adrese (4).

Robert Wilhelm oznámil novou verzi knihovny FreeType 1.1, která slouží k práci s fonty ve formátu TrueType. Knihovnu najdete na adrese (5).

Robert S. Maier (rsm@math.arizona.edu) vytvořil novou verzi balíku GNU plotting utilities (6). Balík je doplněn manuálem o 110 stranách. Bližší informace o balíku naleznete na adrese (7).

Michael Fulbright (msf@redhat.com) oznámil novou verzi RPM balíků grafického prostředí GNOME (8).

Henrik Harmsen (hch@cd.chalmers.se) oznámil vytvoření nové verze souborového manažeru FileRunner, který najdete na adrese (9).

Mark R. Boyns (boyns@sdsu.edu) dokončil další beta verzi svého filtrujícího proxy serveru kompletně napsaného v Javě pod názvem Muffin. Samozřejmostí je podpora HTTP/1.1 a SSL, grafický interface a vzdálená administrace za použití WWW prohlížeče. Jednotliví uživatelé si dokonce mohou napsat vlastní filtrující skripty v Javě. Tento velice zajímavý projekt naleznete na adrese (10).

Společnost Objective Development (11) vytvořila nového klienta pro protokoly používané Microsoft Windows (SMB, CIFS, LanManager) pro unixové operační systémy. S tímto klientem (jmenuje se Sharity) můžete s Windows pracovat stejně, jako by to byly NFS servery. Produkt naleznete na adrese (12).

Hru Freeciv, známou to variantu hry Civilization, si nyní můžete zahrát již ve verzi 1.6.3 — hru naleznete na serveru (13).

DJ Delorie (dj@delorie.com) vytvořil balík karetních her pro X-Window. Komu jméno DJ Delorie něco říká, jistě si balík vyzkouší. Najdete jej na adrese (14).

Photoshop pro Linux. Tak takhle bývá většinou nazýván program GIMP. Verzi 1.0 najdete na adrese (15).

> The gods have blessed it.
> The people can compile it.
> The children can dance in the streets.
>
> ---
>
> Bohové mu požehnali.
> Můžete jej překládat.
> Děti mohou tančit v ulicích.
> *Manish Singh v announci nové verze programu GIMP*

Na adrese (16) naleznete server, jehož jméno zní The Linux business applications. Jistě si domyslíte, co tam najdete.

Tim Mann (mann@src.dec.com) doprogramoval novou verzi programu `xboard` (17). Nemnoho dní poté zveřejnil Dušan Dobeš novou verzi svého šachového programu `phalanx`, který najdete na adrese (18). Právě jsem mu dopl-

nil i dvě nové knihovny zahájení, které z něj dokáží udělat opravdu mistra. ■

1 KDE bindings pro jazyk Python
http://theorie.physik.uni-wuerzburg.de/~hoelzer/pub
2 Linuxové konference
http://www.linuxrx.com/Lists/Lists.perl
3 Maxwell
http://www.eeyore-mule.demon.co.uk/
4 VDX 1.2
http://www.bredex.de/EN/vdx/
5 FreeType 1.1
http://www.freetype.org
6 GNU plotting utilities
ftp://ftp.gnu.org/pub/gnu/plotutils-2.1.1.tar.gz
7 GNU plotting utilities
http://www.gnu.org/software/plotutils/plotutils.html
8 GNOME
http://www.gnome.org
9 FileRunner
http://www.cd.chalmers.se/~hch/filerunner.html
10 Muffin 0.7
http://muffin.doit.org/
11 Objective Development
http://www.obdev.at/
12 Sharity
http://www.obdev.at/Products/Sharity.html
13 Freeciv 1.6.3
http://freeciv.ultraviolet.org
14 The Ace of Penguins
http://www.delorie.com/store/ace/
15 Gimp
http://www.gimp.org
16 The Linux business applications
http://www.m-tech.ab.ca/linux-biz
17 Xboard 4.0.0
ftp://prep.ai.mit.edu/pub/gnu/
18 Phalanx
ftp://ftp.math.muni.cz/pub/math/people/Dobes/

Co nového na sunsite.unc.edu?

Pavel Janík ml., 1. července 1998

X11

X11/screensavers/xscreensaver-2.24.tar.gz — screensaver ala After D**k

X11/xutils/xgreekeychanger-1.3.0.tar.gz — změna klávesových map pro X-Window (xmodmap)

apps

apps/editors/X/GXedit-1.05.tar — Jednoduchý editor založený na knihovně GTK

apps/graphics/viewers/X/imagesort-1.1.tar.gz — třídící program pro obrázky pro X-Window

devel

devel/lang/c/ctags-2.1.tar.gz — generátor značek pro zdrojové texty v jazyce C (pomocí `ctags` je např. vytvořen Linux Source Index (1))

devel/lang/objc/nproducer-1.1.1.tar.gz — konvertor Smalltalk-80 do Objective-C

devel/lang/objc/producer.tar.gz — kompilátor z jazyka Smalltalk-80 to Objective C

games

games/arcade/sf-0.41.zip — hra pro více hráčů

games/strategy/phalanx-16.tar.gz — šachový program pro Linux

hardware

hardware/daemons/jsr_daemon-1.2.tar.gz — pomocí joysticku můžete rebootovat počítač

system

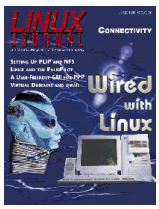
system/network/file-transfer/BeroFTPD-1.0.7.tar.gz — FTP server založený na `wu-ftpd`

■

1 Linux Source Index
http://lsd.linux.cz/LSDIndex

Linux Journal

Petr Bárta, 7. července 1998



V červencovém vydání Linux Journalu je hlavním tématem problematika zapojení do sítí, a to z mnoha různých pohledů. V prvním z takto zaměřených článků se L. Renggli věnuje spojení dvou počítačů pomocí paralelní linky (PLIP) a nastavení NFS — přirozeně, že jde o nastavení v Linuxu.

Jak nastavit jednoduchý router pro připojení LAN sítě do Internetu je obsahem článku M. Hughese.

Pokud spravujete počítačovou síť, můžete si zjednodušit práci při spravování její konfigurace. Stačí použít *Network Information Server (NIS)*, známé také jako *Yellow Pages* (žluté stránky). Jak si poradit s jejich nastavením radí P. Brown.

Popis programového balíku *qmail* — jedné z náhrad `sendmailu` — a jeho nastavení pro příjem pošty pro virtuální domény přináší M. Thomas.

N. Meyers seznamuje se svým programem — grafickým rozhraním pro vytáčení a sledování PPP připojení.

Jak monitorovat síť složenou z různých operačních systémů, a to s využitím Javy, se dozvíte v článku věnovaném balíku *MTool*, jehož autory jsou A. Sostaric, M. Gabor a A. Gygi.

CODA je distribuovaný filesystem vyvinutý na Carnegie Mellon University. Pěkný článek o principech jeho činnosti, jeho vlastnostech atd. přináší P. J. Braam.

Virtuální rozhovor s dvěma z autorů *Samby*, Jeremy Allisonem a Andrew Tridgelem, sestavil J. Blair.

T. Mancill publikuje své zkušenosti s použitím Linuxu

jako routeru pro WAN síť a vyzdvihuje jeho výhody před použitím „BigNames“ routerů.

I v tomto čísle Linux Journalu je mnoho reklam, několik seznámení s novými knihami a distribucemi (i když tady se přece jen projevuje několikaměsíční doba od uzávěrky), a několik dalších článků, ze kterých mne zaujal např. článek o propojení PalmPilota s Linuxem. ■

Fax na trojaký spôsob

Ing. Róbert Dobozy, 9. júla 1998

V tejto kuchárke si povieme niečo o tom, ako pripraviť pre našich používateľov server, ktorý im umožní posielať, prijímať a prezerať faxy. Krok po kroku si povieme ako skompilovať, nainštalovať a nakonfigurovať príslušný softvér. Táto kuchárka je určená hlavne pre tých, ktorí ešte nemajú faxový server nainštalovaný, ale niektoré perličky tu hádam nájdu aj ostrieľaní „faxisti“.

Takže čo potrebujeme na prípravu faxového servera? Je to samozrejme linuxový (unixový) server, class 2 alebo 2.0 (veľmi dôležité, viď. ďalej) faxmodem, balíky `mgetty+sendfax` (1), samba (2), `respond+printfax.pl` (3) a ghostscript (4). Keď sa nám podarí všetky tieto veci spolu nainštalovať a nakonfigurovať, tak dostaneme plnohodnotný faxový server, ktorý nám umožní posielať faxy priamo z Unixu ale aj z prostredia MS Windows alebo MacOS. Ďalej budeme môcť faxy prijímať, tlačiť, prezerať, opakovane posielať, archivovať a ďalej spracovávať. Pozn.: Všetky príklady (napr. cesty k súborom) sú robené na RedHat Linuxe 4.2 a 5.0. Preto je možné, že vo vašom Debiane alebo Slackware to bude mierne inak, ale princíp zostáva zachovaný.

Ako to vlastne funguje? Prijímanie faxov je vcelku jednoduché a prebieha nasledovne. Keď k nám niekto zavolá a na druhej strane telefónnej linky je fax, náš faxmodem to rozpozná (toto je jeden z kameňov úrazu) a povie to programu `mgetty`. Ten zabezpečí, aby sa dané faxové dáta prijali, skontrolovali a po stranách uložili na určené miesto v podobe unixových súborov. Potom `mgetty` spustí vopred určený program (defaultne `/usr/local/bin/new_fax`), ktorý zabezpečí ďalšie spracovanie (napr. vytlačenie, založenie do archívu) prijatého faxu. Posielanie faxov je trochu zložitejšie, pretože tu je viacero možností ako to urobiť. Najprv si samozrejme musíme pripraviť stránky, ktoré chceme odfaxovať. Tieto musia byť v niektorom z podporovaných formátov (ps, ascii text, dvi, pbm, pgm, ppm, g3, lj — HP LJ PCL 4 alebo xwd). Keď ich máme, tak ich prostredníctvom príkazu `faxspool` zaradíme do spoolu, kde si počkajú na odoslanie. Toto bolo na úrovni unixu. Ak chceme faxovať z Windows, tak sa nám to začína komplikovať. Môžme na to využiť SAMBU, kde urobíme tlačiareň na ktorú môžu tlačiť všetky programy, alebo pokiaľ máme Win NT alebo iný systém s podporou unixových tlačiarní (`lpd`) môžme tlačiť priamo na unixovú tlačiareň, ktorá sa tvári ako fax. My sa budeme zaoberať prvým prípadom, pretože je viac používaný a vcelku ľahko implementovateľný. Takže z nejakého windovsového programu vytlačíme stránky na sambovú tlačiareň, samba to predá príslušnému konverznému programu a potom sa to príkazom `faxspool` naspooluje a pripraví na odoslanie. Potom sa používateľovi pošle prostredníctvom winpopu správa o tom, že fax bol daný do spoolu. To je všetko pekné, ale niečo tu chýba, nie? Ano, je to telefónne/faxové číslo prijímateľa. V unixe to je jednoduché, dáme ho ako parameter príkazu `faxspool`. Ale čo Windows? Tu máme zasa dve možnosti. Buď zakomponujeme tel. č. prijímateľa do textu faxu tak, aby nebolo viditeľné (čo napr. v MS Worde nie je problém) a v konverznom programe toto číslo vyextrahujeme. Alebo použijeme špeciálny program s názvom `respond`, ktorý beží na pracovnej stanici a ktorý si na príkaz konverzného programu vypýta meno a telef. číslo prijímateľa a tieto doplní do volania príkazu `faxspool`. Nás zaujíma druhá varianta, pretože je systémovejšia a transparentnejšia ako prvá. V prvom prípade predanie tel. č. silne závisí od použitej aplikácie. Predstavte si, že chcete odfaxovať obrázok napr. z programu PaintBrush alebo Corel. Neviem si predstaviť, kam by ste vložili tel.č. tak, aby bolo parsovateľné konverzným programom a zároveň neviditeľné v obrázku. Teraz to zhrniem. Máme tlačiareň v sambe, na ktorú tlačíme faxy, samba spustí konverzný program, ktorý si prostredníctvom démona `respond` na vašej pracovnej stanici vypýta tel. číslo a identifikáciu prijímateľa a s týmito údajmi potom zavolá príkaz `faxspool`, ktorý — ako som už napísal — zaradí stránky do spoolu, kde počkajú, kým ich niekto odošle. Ale kto? Tak na to je tu program `faxrunq` alebo jeho prefíkanejší brat `faxrunqd`. `Faxrunqd` beží ako démon a raz za stanovené obdobie prezrie spool a prostredníctvom programu `sendfax` sa pokúsi odoslať všetky pripravené faxy (preto sa balík volá `mgetty+sendfax`, to aby ste si nemysleli, že `sendfax` tam je len na parádu a že nič nerobí). Keď je všetko dokonané, tak `faxrunqd` môže spustiť success alebo fail program (podľa toho, či fax úspešne odišiel alebo neodišiel vôbec), v ktorom si môžeme nadefinovať vlastné činnosti. My to používame na archiváciu odoslaných faxov. O neúspešnom odoslaní resp. neodoslaní faxu je informovaný fax administrátor, prípadne aj odosielateľ faxu (viď. poslednú časť článku).

Inštalácia a konfigurácia mgetty+sendfax

Poviete si: načo by som to inštaloval a kompiloval, keď mám rpm-ko? Je na to viacero dôvodov. Po prvé, je možné, že novšia verzia podporuje aj váš nový modem. Po druhé, máte tam kompletnú dokumentáciu v rôznych formátoch. Po tretie, máte tam rôzne pekné alebo menej pekné nástroje, príklady a pomocné balíky. No a nakoniec, niektoré parametre sa dajú nastaviť len v čase kompilácie. Našťastie to zvyčajne nie sú kritické parametre a minimálne RedHati (iné, napr. `deb` balíky som nemal možnosť vyskúšať) ich nastavujú na celkom rozumné hodnoty. Pokiaľ som vás presvedčil o potrebe stiahnutia balíku urobte tak (nemusíte ho kompilovať, stačí keď si pozriete dokumentáciu a pomocné utility a potom si nainštalujete rpm-ko). Pokiaľ som vás nepresvedčil, tak kľudne skočte na časť s názvom konfigurácia. Predtým si ešte pozrite aspoň online nápovedu na (5) alebo na (6).

Inštalácia

Ako prvé balík samozrejme rozbalíme do vhodného adresára (príkazom `tar -xvzf`). Ako druhý krok si pozrieme súbor `README.1st` (vždy čítate readme, že `:-)`. Tam sa hneď na začiatku dozvieme odkazy na online nápovedu, ktoré som spomenul o odstavec vyššie. Postup uvedený v tomto súbore si však mierne upravíme. Ako tretí krok si skopírujeme súbor `policy.h-dist` do súboru `policy.h`. Zatiaľ nič nemeníme. Potom vojdeme do adresára `doc` (`cd doc`) a urobíme `make doc_all`, čo nám vygeneruje komplet-

nú dokumentáciu. Pokiaľ chceme len HTML verziu, nahliadnutím do `Makefile` zistíme, že stačí urobiť len `make mgetty.html`. Na to, aby sme zo súboru `mgetty.texi` dostali napr. HTML verziu musíme mať nainštalovaný balík `tetex-*.rpm`. Pokiaľ ho nemáme a ani nemienime inštalovať, tak si pozrieme online verziu. Teraz doporučujem skopírovať si zaujímavé súbory do iného adresára a urobiť `make fullclean`. Toto nám zmaže vygenerovanú dokumentáciu (máme ju skopírovanú v inom adresári) a uvedie adresár `doc` do pôvodného stavu. Veľmi doporučujem, aby ste si dôkladne prečítali v tej skopírovanej dokumentácii v časti „Common problems and solutions“ sekciu pre váš modem a použitý operačný systém. Ďalej doporučujem prezrieť si `modem.db` a veľmi zaujímavé čítanie je aj `ttyS-cua.txt`. Keď sme všetko pochopili, môžeme pristúpiť k editácii súboru `policy.h` a podľa nadobudnutých vedomostí nastaviť parametre pre váš modem. Okrem parametrov pre modem sú zaujímavé aj nasledovné premenné:

- DEVICE_GROUP — defaultne je skupina „modem“, ktorá v RedHate neexistuje, preto som tam dal „uucp“
- SYSLOG — či sa má použiť `syslog` na zaznamenávanie prevádzky `mgetty` (doporučujem odpoznámkovať),
- MGETTY_PID_FILE — v popise parametra je až do verzie 1.1.15 chyba, má tam byť `%s` a nie `%d` t.j. napr. pre Red Hat má tento parameter vyzerať takto: `/var/run/mgetty.%s`
- FAX_STATION_ID — tel. č. faxu, z ktorého chcete odosielať faxy (bude uvedené v záhlaví stránok)
- FAX_DIAL_PREFIX — kód, ktorý sa má poslať modemu pri vytáčaní tel. č. pred samotným číslom. Potrebné napr. keď musíte vytočiť nulku pred tým ako dostanete normálny oznamovací signál alebo vtedy keď ste pripojený ešte na starú analógovú tel. ústredňu, ktorá vie len pulznú voľbu (použite ATDP)
- FAX_MODEM_TTYS — názvy zariadení, na ktorých máte faxmodemy (dúfam, že ste čítali `ttyS-cua.txt :-)`

Toto boli najdôležitejšie parametre, ktoré je potrebné zmeniť. Ďalej už len v krátkosti vymenujem tie, ktoré som musel meniť alebo aspoň poznať, aby mi to chodilo:

- MODEM_INIT_STRING (podľa vedomostí nadobudnutých štúdiom dokumentácie),
- MODEM_CMD_SUFFIX, MAIL_TO (nechajte default a vytvorte si v `/etc/aliases` alias na vás),
- FAX_NOTIFY_PROGRAM (tento default je použitý aj v skompilovanom RPM balíku od RedHatu).

Keď si myslíme, že sme úspešne dokonfigurovali, tak skúsme obligátne `make`. Malo by to vcelku rýchlo a bezbolestne skompilovať celý balík aj s dokumentáciou. Príkaz `make testdisk` slúži na otestovanie toho, či vie `mgetty` správne zistiť voľné miesto v súborovom systéme. Pokiaľ sa vám nechce, tak ho nemusíte skúšať, lebo na Linuxe to funguje. Na dve nasledovné činnosti budeme potrebovať rootovské oprávnenia. Sú to `make install`, ktoré nainštaluje potrebné súbory do `/usr/local/[bin,sbin,etc,lib]` a editácia `/etc/inittab` tak, aby sa `mgetty` spúšťalo po štarte systému. Ja som vo svojom inittabe použil nasledovný záznam :

```
f1:3:respawn:/usr/local/sbin/mgetty -x 3 ttyS1
```

Mgetty sa nemôže púšťať z príkazového riadku, ale len z `inittabu`, pretože sa počas svojej činnosti reštartuje (a na to potrebuje vlastnosť initu — respawn). Zatiaľ však init nereštartujeme ani `mgetty` nijako nespúšťame, ale pristúpime k dokonfigurovaniu a až následne k otestovaniu balíka `mgetty+sendfax`.

Konfigurácia a testovanie

Konfiguračné súbory vzniknú po inštalácii v adresári `/usr/local/etc/mgetty+sendfax` (keď ste si ho kompilovali sami), alebo v `/etc/mgetty+sendfax` (keď ste použili RedHaťácke RPM-ko). Nachádzajú sa tam súbory `*.config` a `faxheader`. Mgetty, ako ste si určite všimli, vie viac, ako prijímať a posielať faxy. Je primárne určený ako lepší ekvivalent k programom `getty` alebo `uugetty` na pripojenie sa k počítaču modemom. Umožňuje tiež urobiť callback (niekedy v nejakom inom článku) a keď sú správne ponastavované spôsoby zamykania a použité rovnaké znakové zariadenia, vie spolupracovať s `minicomom` a `pppd` (my máme vo firme dial-in, dial-out, callback, ppp a fax a to všetko na jednom modeme a dokonca to aj funguje). A vie to ešte jednu peknú vec. V spolupráci s vgetty (je to súčasťou balíka) umožňuje urobiť z Vášho Linuxu hlasový záznamník s odkazovačom. Ako vidno celý balík je vcelku komplikovaný, a preto sa budeme zaoberať len konfiguráciou jeho faxovej časti. Sú to všetky súbory obsahujúce v názve slovo `fax (:-)` a `mgetty.config`. Pre všetky platí to, že pokiaľ ste celý balík kompilovali sami, tak sa tam použili ako default hodnoty zo súboru `policy.h`, a teda potrebné zmeny budú minimálne. Pokiaľ nie, pozrite si ešte raz online dokumentáciu, kde nájdete správne nastavenia potrebných parametrov. No a teraz si ich (tie súbory a parametre) pekne zaradom preberme. Ako prvý skúsme `mgetty.config`. Tu je dôležité mať správne nastavené `fax-id` a `speed`. Potom pre konkrétne zariadenie a konkrétny modem je potrebné nastaviť jednotlivé sekcie napr. `port ttyS1` a v nej hlavne `init-chat`.

Druhý dôležitý súbor je `sendfax.config`. Svojou štruktúrou je veľmi podobný predchádzajúcemu súboru. Zaujímavé parametre sú tieto: `fax-devices` (jeden alebo viac faxmodemov, cez ktoré sa môžu posielať faxy, oddelené dvojbodkou [:]), zasa `fax-id`, `dial-prefix` (ATDP pre analógové ústredne t.j. pulznú voľbu) a pre každý port (modem) osobitne jeho špecifické vlastnosti, napr. `modem-handshake`.

Teraz je na rade `faxrunq.config`. Tento súbor je konfiguračný pre program `faxrunq` ale aj pre `faxrunqd` (ktorý kedy použiť, viď nižšie). Tu sa nastavuje napr. či sa má faxadminovi posielať mail s informáciou o úspešnom/neúspešnom odoslaní faxu (pokiaľ na to chcete použiť sambu a nie unixový mail, pozrite si poslednú časť článku) a čo je asi najzaujímavejšie `success-call-program` a `failure-call-program`. Tieto, ako z ich názvu vyplýva, sa spustia po úspešnom odoslaní alebo po neodoslaní faxu. Návod na to, ako ich použiť na archiváciu odoslaných faxov je zasa v poslednej časti článku. Takže v tomto súbore zatiaľ nič nemeníme. Zatiaľ posledný súbor je `faxheader`. Toto je záhlavie faxu, ktoré sa doplní na každú odoslanú stranu. Je tam vhodné napísať `FROM: <názov_vašej_organizácie>` a správne faxové číslo (ak ste balík kompilovali sami, tak by tam malo byť). Prichádza hodina h minúta m a sekunda s, kedy si otestujeme, čo sme doteraz spáchali. Skúsme spraviť `init q`, čo spôsobí, že

```
unicorn/tmp.56 >faxspool -F robo -f robo 07273053
/tmp/fax.g3 Neither /usr/local/etc/mgetty+sendfax/fax.allow nor
/usr/local/etc/mgetty+sendfax/fax.deny exist, so only root may use the
fax service. Sorry.
```

Výpis č. 1: Faxspool security

si `init` prečíta `inittab` a spustí nám `mgetty`. V adresári `/var/log` by mal vzniknúť súbor `mgetty.<názov_tty>`. Pokiaľ tento súbor nevznikol alebo je prázdny, tak vám `mgetty` nemá čo povedať a teda všetko je v poriadku. Ešte sme však nevyhrali! Ak máte `minicom` alebo používate PPP, skúste ich. S popísanou konfiguráciou by to malo chodiť. Ak máte ešte jeden modem, skúste z neho zavolať (cez `minicom`) na váš fax. `Mgetty` by mal normálne hodiť login a umožniť vám prihlásenie. Teraz otestujeme prijímanie faxov. Nechajte si od niekoho poslať krátky fax na váš faxmodem. Ako som už napísal, mal by v adresári `/var/spool/fax/incoming` vzniknúť súbor začínajúci písmenom `f`. Po prijatí faxu sa `mgetty` pokúsi spustiť program `FAX_NOTIFY_PROGRAM` (definovaný v `policy.h`, čo je štandardne `/usr/local/bin/new_fax`), s nasledovnými parametrami:

```
<návratový_kód> '<identifikácia_odosielatel'a>'\
<počet_strán>\
<meno_prvého_súboru> <meno_druhého_súboru> ..
```

V inštalačnom adresári `mgetty` nájdete adresár `samples` a v ňom množstvo súborov s názvom `new_fax.*`. Napr. `new_fax.lj` vytlačí prijatý fax na definovanej tlačiarni (typu HP LJ). Ja však odporúčam pozrieť si adresár `new_fax.all`, kde je pekný balík modulov, ktoré vedia robiť s prijatými faxmi rôzne zaujímave veci (napr. vytlačiť, odfaxovať ho ďalej, poslať informáciu o prijatom faxe mailom, zápisom do logu). Je to veľmi pekne konfigurovateľné (pozrite súbor `faxlist`). Inštalácia je veľmi jednoduchá. Nahrajte všetky súbory do samostatného adresára a urobte link z `/usr/local/bin` na `new_fax` (alebo to všetko nahrajte priamo do `/usr/local/bin`). V `new_fax` sú nejaké konfiguračné cesty a vo `faxlist` si nastavíte, ako sa má `new_fax` správať. Keď chcete používať `new_fax` aj na archivovanie faxov, prekúšte sa až k poslednej časti tohto článku. Dajte si ešte raz poslať nejaký fax a už by sa mal spustiť `new_fax`, ktorý by mal vykonať požadovanú činnosť. Všetko chodí? Výborne. Polovičku testovania (`mgetty`) máme za sebou. Skúsme druhú polovičku — `sendfax`. Na to potrebujeme nejakú obeť v podobe iného faxu a niečo, čo tej obeti pošleme. To niečo môže byť hocijaký krátky (!) súbor vo formáte, ktorý ovláda príkaz `faxspool` (viď. časť s názvom Ako to vlastne funguje?). Pokiaľ nemáte poruke vhodný súbor, môžete si ho vyrobiť spôsobom podľa výpisu:

```
pbmtext -font\
    /usr/local/lib/mgetty+sendfax/cour25.pbm \
   >/tmp/fax.pbm
```

Na štandardný vstup napíšeme nejaký text (napr. `AHOJ toto je môj prvý fax`) a stlačením `Ctrl-D` `pbmtext` vygeneruje súbor `/tmp/fax.pbm`. Čo ste vyplodili, si môžete pozrieť napr. programom `zgv` alebo `xv`. Tento pbm súbor skonvertujeme do formátu g3 (štandardizovaný formát, ktorý sa používa na prenos faxov) programom `pbm2g3` (nie `pbmtog3` !!!), ktorý vznikol pri inštalácii balíku `mgetty+sendfax`. Urobte teda

```
pbm2g3 /tmp/fax.pbm > /tmp/fax.g3.
```

Program `pbmtext` ako aj iné konverzné programy (`ppmto*`, `pgmto*`, `pbmto*`, `pnm*`), ktoré používa `faxspool` sa nachádzajú v balíku `libgr-progs` (RH5.0) resp. `netpbm` (RH4.2). Tento pripravený súbor teraz programom `faxspool` pripravíme na odoslanie:

```
faxspool -F <vaše_meno> -f <váš_email> \
   <faxové_číslo_obete> /tmp/fax.g3
```

Keď sa faxspool začne rozčuľovať, že nie ste root Faxspool security, tak je potrebné vytvoriť prázdny súbor `/usr/local/etc/mgetty+sendfax/fax.deny`. Tento súbor slúži na to, aby sa dalo obmedziť, kto môže a kto nemôže posielať faxy. Podrobnejšie viď poslednú časť článku. Keď je `faxspool` spokojný, mal by povedať niečo o tom, že fax bol naspoolovaný a že `faxrunq` nebol spúšťaný počas posledných 24 hodín (čo nie je problém, lebo sme ešte neskončili konfiguráciu). Ako to vyzerá vidieť na výpise Ukážka spoolovania faxu. Ak všetko prebehlo v poriadku, tak v adresári `/var/spool/fax/outgoing` vznikol adresár `F<XXXXXX>`, kde `<XXXXXX>` je nejaké číslo. V tomto adresári sa nachádza súbor `JOB` s informáciami o danom faxe a jeden alebo viac súborov `fX.g3`, ktoré obsahujú jednotlivé stránky faxu. Na prezeranie obsahu faxového spoolu slúži príkaz `faxq` a na vymazanie zákazky zasa `faxrm`.

Pozrime sa teraz na to, ako odoslať pripravené faxy. Máme dve možnosti. Buď použijeme program `faxrunq`, ktorý sa bude v pravidelných intervaloch spúšťať z `cronu`, alebo pri štarte systému spustíte program `faxrunqd` ako démon a ten v pravidelných intervaloch (60 s) prezrie spool a rovnako ako `faxrunq` pomocou programu `sendfax` pošle pripravené faxy prijímateľom. Ktorý z nich použiť? `Faxrunq` je napísaný v shelli a spúšťa sa raz za čas, zaberá teda menej systémových zdrojov. `Faxrunqd` beží stále, je napísaný v Perle, má ale viac možností (napr. vie obsluhovať viac modemov, vie spracovávať faxy podľa priority), je flexibilnejší a má rýchlejšiu odozvu. Z tohto dôvodu doporučujem použiť `faxrunqd`. Urobte si rc skript (napr. `/etc/rc.d/rc3.d/S96fax` — RH alebo do `/etc/rc.d/rc.local` — Slackware), v ktorom sa bude spúšťať `faxrunqd` s parametrom `-l tty<n>`, kde `<n>` je príslušné zariadenie s modemom. Pokiaľ chcete použiť viac zariadení, tak ich oddeľte dvojbodkou (:). Na otestovanie toho či nám `faxrunqd` a `sendfax` správne fungujú s naším modemom (zatiaľ máme otestovanú len polovičku, pamätáte?), spustíme `faxrunqd` (ako root) z príkazového riadku: `faxrunqd -l ttyS1`. V adresári `/var/spool/fax` vzniknú dva súbory — `acct.log` a `faxrunqd.log`. V týchto súboroch sa nachádzajú informácie o tom, ako dopadlo posielanie faxov. Keď fax úspešne odišiel a je dokonca v mieste prijatia čitateľný, máme unixovú časť odosielania vybavenú.

Ako som už spomenul v časti Konfigurácia a testovanie, pri spoolovaní faxu vznikne súbor `JOB`, v ktorom sú uložené rôzne informácie o faxe. Program `faxrunqd` si ich pred odoslaním prečíta a na súbor `JOB` vytvorí hardlink s ná-

```
unicorn/tmp.57 >faxspool -F robo -f robo 07273053 /tmp/fax.g3
spooling to /var/spool/fax/outgoing/F000017...
spooling /tmp/fax.g3...
/tmp/fax.g3 is format: g3

Putting Header lines on top of pages...

Fax queued successfully.

WARNING: faxrunq hasn't been run in the last 24 hours.
         Faxes only get sent out when faxrunq runs! Contact Fax
administrator.
```

Výpis č. 2: Ukážka spoolovania faxu

zvom `JOB.locked`, čo je vlastne veľmi jednochuchý spôsob zamykania a ochrany. Ak sa mu poslanie podarilo, tak súbor JOB premenuje na `JOB.done`. Ak nastala nejaká závažná chyba, tak vznikne `JOB.error`, a keď sa mu nepodari fax odoslať (obsadená linka, na druhej strane nie je fax), tak `JOB.suspended`. Keď `faxrunqd` ukončí svoj beh, tak samozrejme `JOB.locked` unlinkuje (zruší). Ani jeden fax so statusom `.suspended`, `.error` alebo `.done` už naďalej nie je spracovávaný. Preto je pri statusoch `.suspended` a `.error` potrebné zistiť príčinu chyby. S týmto súvisí aj správanie programu `faxq`. Tento štandartne zobrazuje len faxy pripravené na odoslanie. Preto je potrebné z času na čas skontrolovať stavy faxov príkazom `faxq -a` (čo zobrazí všetky, aj suspendnuté faxy). Zistiť príčinu chyby je možne použitím `faxq -a -v`. Pokiaľ bol fax len suspendnutý, je možné ho znova zaradiť do spracovania pomocou `faxq -r` (môžte na to urobiť napr. `cron` job). Faxy so statusom `.error` je potrebné dať do poriadku ručne (opravením údajov v súbore `JOB.error` a jeho premenovaním alebo zmazaním a opätovným poslaním apod.).

```
;
; Add to your smb.conf
; for use with printfax.pl V1.3
;
[fax]
   comment       = Fax
   postscript    = yes
   print command = ( /usr/bin/printfax.pl %I %s\
                     %U %m; rm %s ) &
   printable     = yes
   writable      = no
   path          = /your/samba/print/dir
;
; Add your own security options!!!
;
```

Výpis č. 3: Definícia faxu v sambe

Čo v prípade, že niečo nefunguje? Je ťažké dať nejakú všeobecnú radu. Musíte skúmať logy, zvýšiť debug level (parameter `-x` pre `mgetty` a `debug` v `sendfax.config` pre `sendfax`), prečítať si ešte raz online dokumentáciu a skúsiť nejakú inú variantu inicializácie príp. ovládania modemu. Presvedčte sa, že váš faxmodem je naozaj triedy 2.0 alebo aspoň 2 (to nie je to isté). Bohužiaľ `mgetty` zatiaľ nepodporuje faxmodemy triedy 1 (ja som vás na začiatku varoval). Doering (autor `mgetty`) vraj na tom priebežne pracuje, ale je to dosť komplikované a trvá to dlho (pretože mnoho vecí, ktoré pri class 2/2.0 urobí modem je potrebné urobiť v programe). Ďalej sa presvedčte, či máte v modeme správnu verziu firmware, ktorá naozaj podporuje class 2/2.0 faxy. Ako som sa presvedčil na vlastnej koži, to že to je napísané na krabici alebo v reklamných materiáloch neznamená, že je to pravda. Keď niektorí výrobcovia modemov zaviedli podporu pre voice, tak vyhodili podporu pre faxy. Je to bohužiaľ niekedy tak trochu čierna mágia a je to asi najväčšie úskalie, aké vás čaká. Preto som toľko opakoval, že si máte dôkladne prečítať dokumentáciu. Keď si už naozaj nebudete vedieť poradiť, tak napíšte buď priamo Doeringovi (jeho adresa je v `README.1st`), ale pred tým si však prečítajte BUGS. Alebo skúste mailing list, ktorého archív je na adrese (7).

Ak to nakoniec funguje, tak máme vyriešenú a sfunkčnenú unixovú časť problému. Pozrime sa na to, ako to vyzerá s naviazaním na MS Windows.

Inštalácia a konfigurácia respond a printfax.pl

Stiahnite si binárku programu `respond.exe`, súbor `printfax-1.3.5.pl` (alebo vyššiu verziu, ak medzitým vznikla), `readme.txt` (prečítať) a voliteľne aj `faxlpq` a `faxlprm`. Začnime od konfigurácie unixu. Predpokladá sa, že máte nainštalovanú a funkčnú sambu a perl 5.x. Skopírujte súbor `prinfax-*.pl` do `/usr/local/bin/printfax.pl` a do `smb.conf` (zvyčajne v `/etc`) pridajte definíciu tlačiarne s názvom `fax` tak, ako to je na výpise Definícia faxu v sambe. V `printfax.pl` je pre Linux ešte nutné zmeniť premennú `$smbclient` tak, aby ukazovala na plnú cestu k programu `smbclient` (`/usr/bin/smbclient`), a ak ste na inštaláciu `mgetty+sendfax` použili RedHaťácke RPMko tak aj premennú `$faxspool` na `/usr/bin/faxspool`. Tým by mala byť unixová časť hotová a poďme sa pozrieť na zúbok Windowsom. Tam nastavte, aby sa pri štarte spúšťal winpopup (na prijímanie správ od faxového systému) a `respond.exe`, ktorý sme pred tým nakopírovali do nejakého inteligentného adresára (ja to mám v `c:/utils/fax`). Teraz sa pozrime do Network Neighbourhood (Síť, či ako to je v českých Windowsoch), alebo skúsme Start/Run a tam napíšte

```
\\smb_meno_servera_s_faxom
```

a mali by sme tam vidieť tlačiareň s názvom fax (definované v `smb.conf`). Dvojklikom na ňu Windowsy zistia, že ju nemáme nainštalovanú a opýtajú sa, či ju chceme nainštalovať. Áno chceme. V ďalšom okne si musíme vybrať vhodnú tlačiareň. Ide o postScriptovú tlačiareň, tak si vyberte napr. HP LaserJet 5P/5MP PostScript alebo IBM 4019

LaserPrinter PS17 a nainštalujte ju. Potom tlačiareň premenujte na fax a pozrite si jej vlastnosti. Dôležitá je záložka Fonts, kde musí byť zakliknuté, že Send TrueType fonts to printer, ostatné položky by mohli zostať na svojich default hodnotách. Ubezpečte sa, že beží winpopup a `respond` a skúste vytlačiť nejakú stranu na fax. Keďže fax je postscriptová tlačiareň, tak ovládač tlačiarne zabezpečí, že sa daná stránka prekonvertuje na viac či menej kompatibilný a prenositeľný Postscript (závisí od ovládača tlačiarne) a pošle sa na sambovskú tlačiareň. Tam sa toho ujme `printfax.pl`, ktorý otvorí spojenie na program `respond`. Tento sa maximalizuje a vypýta si od používateľa meno a faxové č. prijímateľa ako aj meno používateľa. Tieto údaje sú potom poslané naspäť `printfaxu`, ktorý na ich základe pomocou `faxspool` vygeneruje požiadavku na fax. `Faxspool` použitím `ghostscriptu` prekonvertuje Postscript na g3 a vyrobí spoolovú požiadavku. Toto je už prípad, ktorý sme prebrali v predchádzajúcej časti a ktorý nám už chodí. Jediný problém vidím v tom, že odoslaný fax bude zle sformátovaný alebo `ghostscript` nebude vedieť správne vygenerovať g3 súbor (napr. chýbajúce fonty). Potom pomôže už len skúšať rôzne typy postscriptových ovládačov (tlačiarní) a ich rôzne nastavenia. Voliteľne môžte ešte pre fax v `smb.conf` zadefinovať `lpq` command (cesta k `faxlpq`) a `lprm` command (cesta k `faxrm`).

Teraz sme v situácii keď faxy prichádzajú, odchádzajú, Windowsy spolupracujú, no proste znie to neuveriteľne, ale je to hotové a funkčné. Pokiaľ ste spokojní, môžete kľudne prejsť k ďalšiemu (iste zaujímavému) článku Linuxových novín. Ak vás však zaujímajú podrobnosti, prípadne chcete funkcionalitu faxového servera ešte vylepšiť, tak sa so mnou v treťom kole ponorte do problému ešte hlbšie.

A poďme ešte hlbšie. Tu už budú informácie kusejšie a nebudú na seba tak nadväzovať, pretože sa budeme venovať rôznym perličkám v rôznych častiach faxového systému. Najprv si povedzme nejaké všeobecnejšie informácie: g3 (group 3) je všeobecne uznávaná a používaná norma (CCITT) na prenos dát medzi faxmi. Existujú dva druhy g3 súborov. S vysokým rozlíšením 204x196 dpi a s nízkym (normálnym) rozlíšením 204x98 dpi. Keď sa tieto rozlíšenia pomiešajú, tak stránky faxu budú buď o polovičku kratšie alebo 2 krát dlhšie. `Mgetty+sendfax` vie spracovávať obidva typy rozlíšení. Jedna strana faxu má zvyčajne šírku 1728 pixelov a dĺžku (A4) 2100 pixelov. Ak máte v systéme nainštalované pbm utility, tak sú medzi nimi aj `pbmtog3` a `g3topbm`. Niekde v časti Inštalácia a konfigurácia `mgetty+sendfax` som spomínal, že na konverziu medzi pbm a g3 máte používať `pbm2g3` (je to v balíku `mgetty`) a nie `pbmtog3`. Že prečo? Jednak `pbm2g3` a `g32pbm` sú rýchlejšie ako pôvodné programy a jednak `pbm2g3` produkuje g3 dáta, ktoré spĺňajú štandard T.4, čo nám umožní bezproblémové posielanie faxov.

Určite ste si počas konfigurácie inicializácie modemu všimli, že modem musí mať vypnutý autoanswer (`ATS0=0`). Linku zdvíha priamo `mgetty` (parameter `rings`). Príčinou tohto správania okrem iného je, že je to vlastne jediná možnosť, ako rozlíšiť, či prichádzajúce volanie je fax alebo dáta (čo môže byť súčasťou čiernej mágie konfigurovania modemu). Keď to modem nevie správne povedať, je jediná možnosť nastaviť modem a `mgetty` natvrdo do dátového alebo faxového módu. Slúži na to parameter `modem-type`. `Sendfax` (aj `faxspool`) vedia s niektorými faxmodemami robiť tzv. fax polling. Je to, keď vy niekomu zavoláte a príjmete dokument. Je to popísané v online dokumentácii v časti Fax Operations.

Ďalšou zaujímavou vlastnosťou je možnosť použiť externý fax ako skener. Je to popísané v online dokumentácii v časti Fax Operations a v súbore `doc/scanner.txt`.

Niektoré telefónne spoločnosti (slovenský a ani český telekom rozhodne nie `:-)`, možno až budeme mať všetci ISDN) ponúkajú službu *caller-id*. Je to služba, keď je k nám po prvom zazvonení prenesená identifikácia volajúceho účastníka. `Mgetty` vie túto informáciu využiť a na základe caller-id prijať alebo neprijať fax.

Pokiaľ je modem vypnutý, tak sa `mgetty` ukončí (pretože kontroluje funkčnosť modemu a loguje to). Keďže máme v `inittabe` respawn, tak init po chvíli zahlási:

```
INIT Id "f1" respawning too fast:\
        disabled for 5 minutes.
```

Nie je to vážna chyba, ale znamená to, že máte niečo s modemom a asi nijaké faxy nepošlete ani nepríjmete.

Preberme si niektoré zaujímavé vlastnosti a možnosti programu `faxspool`. Ako bolo na začiatku spomenuté `faxspool` vie podľa prípony zistiť typ súboru a potom ho skonvertovať do g3 formátu. Pokiaľ sa mu to nepodarí, pokúsi sa uhádnuť (použitím magic numbers) tento typ.

`Faxspool` umožňuje použiť telefónny adresár. Buď globálny (`/usr/local/etc/mgetty+senfax/faxaliases`) alebo individuálny (`$HOME/.faxnrs`). Formát súborov je rovnaký: `<alias> <faxové_číslo>`. Faxspool najprv pozrie individuálny a potom globálny tel. adresár.

Ďalej nám umožňuje povoliť/zakázať kto môže alebo nemôže posielať faxy. Slúžia na to súbory `/usr/local/etc/mgetty+sendfax/fax.allow` a `/usr/local/etc/mgetty+sendfax/fax.deny`. Pokiaľ ani jeden neexistuje, tak posielať faxy môže len root. Ak existuje len `fax.deny`, tak môžu všetci okrem tých čo sú tam vymenovaní. Ak existuje len `fax.allow`, tak môžu (myslím posielať faxy) len tí, čo sú tam vymenovaní. Tento bezpečnostný model je implementovaný len v programe `faxspool`, ktorý samozrejme nie je setuid root a `/var/spool/fax/outgoing` je zapisovateľný pre všetkých. Preto je veľmi jednoduché tento model obísť napríklad tým, že si tento program niekto skopíruje a kontroly odtiaľ vyhodí. Toto zabezpečenie sa samozrejme týka len unixových používateľov. Keď chcete povoliť/zakázať windowsových používateľov, tak musíte použiť nastavenia v sambe.

V súbore `etc/mgetty+sendfax/faxheader` je uložené záhlavie každej strany. `Faxspool` toto záhlavie preparsuje a nahradí niektoré preddefinované znaky inými. Sú to napr.: `@T@` — tel. č. prijímateľa, `@P@` — aktuálna strana, `@M@` — celkový počet strán, `@U@` — meno odosielateľa, `@DATE@` — dátum odoslania. Úplný popis týchto znakov (tokenov) nájdete v manovej stránke programu `faxspool`.

Je dobrým zvykom pred každý fax vložiť tzv. *cover page* (titulnú stranu), kde sú napísané informácie o odosielateľovi, prijímateľovi a prípadne nejaká krátka správa. `Faxspool` toto umožňuje (prepínač `-C`). Skúste `man coverpg`. Popisy spôsobu implementácie príslušného programu (`make.coverpg`) sa nachádzajú v adresári `samples`. Tento program nie je defaultne nainštalovaný a treba to urobiť ručne podľa manovej stránky.

V niektorých verziách `faxspoolu` je zle definovaný príkaz `echo`. Spôsobí to potom škaredý výpis niektorých textov napr. helpu (neinterpretujú sa escape príkazy ako `\t`). V mojom `faxspoole` (verzia 1.1.15) to je na riadku 80. Sk-

úste si vyhľadať prvý výskyt slova `echo`. Malo by tam byť `echo="echo -e"`.

Balíky `respond+printfax.pl` tiež umožňujú používať telefónne zoznamy a aliasy. Keď namiesto tel. č. vložíte `@<niečo>`, tak sa to `<niečo>` použije ako meno súboru vo vašom domovskom adresári (samozrejme v unixe, ale máme sambu tak ho môžeme zdieľať s Windowsami) a jeho obsah sa použije ako zoznam tel. č. prijímateľov. Jednotlivé položky zoznamu môžu byť oddelené čiarkou alebo medzerou. Ak `<niečo>` začína lomítkom (/), tak sa to berie ako absolútna cesta. Podrobnosti a iné vlastnosti nájdete v `readme.txt`, ktoré by ste mali mať už stiahnuté. Na tom istom mieste sa nachádza aj súbor `fprintfax`, ktorý môže slúžiť na faxovanie cez unixovú tlačiareň. Dá sa ako filter do tlačiarne a potom funguje ako `printfax.pl`.

Iné zaujímavé nástroje nájdete v adresároch `contrib` a `frontends`. V `contribe` sú napríklad nástroje na lepšiu integráciu TEXdvi a `faxspoolu`. Vo `frontends` sú rôzne prezerače a pomocné vstupné (na vytváranie dokumentov) a výstupné (na spracovanie poslaných faxov) programy. Je tam napr. nástroj na integrovanie faxovania do Emacsu, mail to fax gateway, X-Window prezerač faxov (`viewfax` — ak posielate faxy len z unixu, odporúčam), WWW prezerač a posielač faxov (adresár `WWW`), iné rozhrania ako `respond` do Windowsov a v adresároch `winword` a `winword2` dva spôsoby (ktoré som na začiatku článku zavrhol ako netransparentné a neuniverzálne), ako faxovať z MS Wordu.

Nakoniec spomeniem ešte niečo z vlastnej tvorby. Môj kolega Pišta porobil nejaké úpravy a skripty, ktoré by mohli byť užitočné aj Vám. Prvá úprava sa týka toho, že `printfax.pl` štandardne cez winpopup pošle informáciu o tom, či bol daný fax daný do spoolu. To je v poriadku. Potom však `faxrunqd` pošle faxadminovi mail (v unixe) o úspechu alebo neúspechu odoslania. Ale používateľ o tom nič nevie a musí sa stále pýtať faxadmina na stav faxu. Preto sme urobili drobné zmeny (týkajú sa programov `printfax.pl`, `faxpool` a `faxrunqd`), ktoré spôsobia to, že sa winpopom pošle na pracovnú stanicu, odkiaľ prišla požiadavka, správa o stave faxu. Druhý nástroj, ktorý sme vytvorili, je WWW archív a prezerač faxov. Umožňuje prezerať, tlačiť a opakovane posielať prijaté a odoslané faxy, ako aj prezerať stav spoolu. Keďže sa mi nepodarilo (zatiaľ) Doeringa presvedčiť, aby tú informáciu o stave faxu (cez winpopup) dal do distribúcie `mgetty`, tak si môžete patche stiahnúť z adresy(8). Na tom istom mieste nájdete aj WWW archív. Tento text bude možné v čase vyjdenia tohto článku (dúfam) nájsť na adrese (9).

Záverom

Podľa môjho názoru je toto riešenie plnohodnotná a finančne nenáročná (ale napriek tomu dokonale funkčná) alternatíva k akémukoľvek komerčnému riešeniu pre rôzne platformy. ■

1 Mgetty+sendfax
ftp://alpha.greenie.net/pub/mgetty/source/1.1/
2 Samba
http://samba.anu.edu.au/samba/
3 Respond+printfax.pl
http://www.boerde.de/~horstf/
4 Ghostscript
http://www.cs.wisc.edu/~ghost/
5 Domovská stránka mgetty
http://alpha.greenie.net/mgetty/index.html
6 Online nápověda k mgetty
http://www.leo.org/~doering/mgetty/index.html
7 Mailing list mgetty
http://www.elilabs.com/mgarc/index.html
8 Patche pro faxování
ftp://ftp.idata.sk/pub/unix/fax/
9 WWW adresa článku
http://www.idata.sk/~robo/fax/

Jak jsme přesouvali Interbase z NT na Linux

Daniel Prynych, 8. července 1998

Nedávno firma Interbase konečně uvolnila svůj hlavní produkt — SQL databázi Interbase (dále jen IB) — pro Linux. Jde o verzi 4.0 pro Red Hat Linux 4.2 a byla dána volně k dispozici ve variantě pro neomezený počet uživatelů. Protože jsme IB již asi rok provozovali (ve verzi pro Microsoft Windows NT Server 4.0), bylo rozhodnuto přejít na verzi pro Linux, který jsme již používali jako souborový (Samba) a tiskový server a to také proto, že s Windows NT jsme nebyli zcela spokojeni.

IB jsme používali pro tvorbu kusovníku. Kusovník je zjednodušeně řečeno seznam všech dílů pro výrobu stroje. Pro každý stroj existuje tabulka variant a každá součást může existovat v několika variantách. Všechny varianty dané součásti mají společnou hlavičku, která obsahuje základní informace jako je číslo dílu, jeho název atd. Každá varianta obsahuje údaje specifické pro danou součást a její variantu jako je číslo výkresu, ČSN, rozměry, váha atd. Jinak je možno používat kusovník i strukturovaně, to znamená, že pod každou součásti leží další součásti, něco jako stromová struktura adresáře. Jak je vidět, bylo nutno vytvořit několik tabulek, které jsou spolu navzájem svázány. Zápis se provádí jak do jedné tabulky, tak i do několika najednou a v tomto případě se musí provést buď do všech najednou nebo ani do jedné. To bylo jen malé odbočení a teď zpátky k IB.

Použili jsme technologii klient/server, přičemž klientské programy máme napsány v Delphi 2@. Proto měla vzniknout tato sestava: jako server bude pracovat IB na Linuxu a klientské programy poběží pod Windows 95 a Windows NT a s IB budou komunikovat za pomoci SQL-Links protokolem TCP/IP.

Přechod z IB na Windows NT na IB na Linuxu nebyl úplně jednoduchý, a to hlavně proto, že s touto konfigurací nebyly u nás ještě zkušenosti. Přecházelo se za plného provozu, ale nakonec byly všechny problémy vyřešeny.

Nejprve byly prováděny testy IB na RedHatu 4.2 (dále jen RH), pro který je IB certifikována, a RH 5.0 na různých počítačích. Samotná IB na obou systémech pracovala bez problémů, byly použity tyto konfigurace: AMD486/100 48MB, což je pod doporučenou hranicí, Pentium 130 64 MB a 2x Pentium Pro 180 128MB. Zkoušky byly prováděny s dodávaným front-end programem `isql` a to jak ze systému, na kterém běžela IB, tak i z vedlejších počítačů programem `isql`. Poté byla za pomoci `gbak` databáze na NT zazálohována, přesunuta na Linux a rekonstruována.

Tento postup je nutno dodržet, výhodou je i to, že se z databáze odstraní nepoužívaná data a položky se srovnají za sebe, což se projeví i na rychlosti. Při používání programu

isql je zajímavé, že pokud se přihlásím jako ROOT, mám automaticky přístupová práva uživatele SYSDBA, (jedná se o uživatele definované v IB, s uživateli v Linuxu nemají nic společného), mohu tedy s databází provádět všechny operace jako bych byl její vlastník. Tento přístup se mně osobně moc nelíbí, neumožňuje totiž důsledně oddělit funkce správce systému a správce databáze. Jako další test byl sestaven v Delphi jednoduchý program, který v cyklu prohlížel jednu z tabulek databáze. Jednalo se o ceník obsahující asi 40 000 řádek a 15 sloupců zahrnujících jak řetězce znaků, tak celočíselné položky a reálná čísla. SQL příkaz byl postaven tak, že se vliv indexů prakticky neprojevil. Tento program byl spuštěn na třech počítačích — dva Win95 a jeden WinNT 4.0 — a to na každém třikrát. IB tentokrát běžela pod RH 4.2 na AMD 486/100, přesto zvládla zpracovávat neustále devět dotazů najednou. Rychlost sice nebyla velká, ale to se v této konfiguraci ani nedalo očekávat.

Dále jsem pokračoval napsáním programu v C, který bral data z textového souboru a na jejich základě opravoval a doplňoval výše zmíněný ceník. SQL příkazy jsem psal přímo do zdrojového textu a k jejich převodu na API IB funkce jsem použil dodávaný preprocesor gpre. Při tvorbě těchto programů je nutno příkaz SET DATABASE umístit na začátek programu, gpre je patrně jednoprůchodový preprocesor a jako první musí znát jméno databáze, aby mohl prohlédnout definice tabulek, nemůžeme ho tedy použít pro překlad programu, který pracuje s databází, ke které momentálně nemáme přístup. Pro tento případ musíme použít přímo příkazy API IB. Každý(é) příkaz(y) je nutno umístit do transakce např.

```
SET TRANSACTION
   SELECT COUNT (JK) INTO :pocet WHERE JK = :jk
COMMIT
```

Pro bližší vysvětlení doporučuji nahlédnout do dodávané dokumentace, a nedejte se odradit tím, že je zde probíráno IB API pro Windows. Překlad na RH 5.0 je trošku odlišný od RH 4.2 — nemůžeme použít

```
gcc něco.c -lgdslib -ldl -lcrypt
```

překladač začne mít námitky

```
ld:warning: libdl.so.1, needed by\
   /usr/lib/libgdslib.so,\
   may conflict with libdl.so.2
```

Musíme přeložit za pomocí knihovny pipe.

```
gcc něco.c -lgds -ldl
```

Rozdíl je v tom, že v druhém případě je navíc spuštěn proces gds_pipe, který pracuje jako prostředník mezi programem a IB.

Na straně klienta bylo nutno provést upgrade BDE na verzi 4.51 a bylo nutno upravit část programu v Delphi 2, protože při použití většího počtu query — v našem případě asi 15 — a znovuobnovení dat prováděné příkazy query.close; query.open; došlo ke ztrátě spojení s databází. Nahrazením query.close příkazem database.closedataset v kritických částech byly problémy odstraněny. Zajímavé je, že při použití protokolu NetBEUI k tomuto problému nedocházelo.

Nyní už mohla být IB definitivně nainstalována na RH 4.2 běžící na 2x Pentium Pro 180 s 128MB paměti. Samotná data sídlí na dvou SCSI 2MB discích. Nepoužíváme zrcadlení disků, ale využíváme možnosti IB, která umí zrcadlit data. Jedenkrát za den je pomocí gbak provedeno zálohovaní dat s týdenní rotací a je spouštěn program, který doplňuje a opravuje údaje v ceníku. Zajímavé je, že tento program původně napsaný v Delphi a běžící na WinNT a komunikující s IB pomocí DBE zpracovával data asi 1 hodinu. Prakticky stejný program psaný v C a používající API IB pracuje asi 5 minut. Vzhledem k tomu že v současné době přistupují k datům čtyři klientské počítače po celou pracovní dobu a tři další jen občas, je server zatížen jen velmi málo. Pokud porovnáme rychlost IB na NT a na Linuxu, tím myslím rychlost odezev pro klientské programy, zdá se prakticky stejná. IB zároveň běží na druhém počítači, kde slouží jako testovací databáze při opravách programů. Celkově mohu říci, že nasazení koncepce klient/server se nám osvědčilo, klientské počítače nezatěžují server zobrazováním dat a běžnými operacemi, obsluha je velmi snadná, naproti tomu server dodává svůj výkon pro operace s databází všem klientům a velmi snadno zajišťuje konzistenci dat.

Firma Interbase dodává pro Linux také modul pro komunikaci s Perlem. Na RH 5.0 ho bez úprav nepřeložíme. Musíme postupovat takto:

1. rozbalíme ibperl-0_5_tar.gz
2. v souboru Makefile.PL a Makefile.a změníme -lpgslib na -lpgs
3. perl Makefile.PL
4. make
5. doporučeno: nahradit na řádcích INSTALL... /var/tmp/perl-root$(PREFIX)/lib/perl... za /usr/lib/perl...
6. make install, pokud nahlásí při instalaci chybu, je nutno provést předchozí bod
7. otestovat — perl test.pl nebo ./test.pl

Soubor IBPerl.pm by se měl nacházet v adresáři /usr/lib/perl5 a soubor IBPerl.so v adresáři perl5/site_perl/i386-linux/auto/IBPerl.

Co říci závěrem? IB je to, co Linuxu zatím chybělo — a to velmi výkonná SQL databáze s triggery a transakcemi, což je dnes pro bezpečné aplikace nutnost, zvlášť pokud se často mění a přidávají data. Na druhé straně front-end isql nevyniká právě přívětivou obsluhou, psql od konkurenčního postgresql se mi zdá vyřešeno lépe. Co mi ještě chybí, je podpora IB v PHP. IB umožňuje Linux nyní nasadit i v oblasti SQL serverů, zvláště ve spojení s Delphi. Jistě, je tu MySQL, ale to nemá transakce, Postgresql atd., ale IB je prostě o něčem jiném, aspoň v současné době určitě. IB také umožňuje menším programátorským skupinám a jednotlivcům snáze vyvíjet aplikace klient/server, protože nyní je možno si velmi snadno a relativně lacino k vývojovému systému pořídit legálně SQL server s běžící IB, vlastně jen za cenu „železa“, a většinu chyb a problémů, ke kterým dochází při nasazení u uživatele, odstranit už při testech. ■

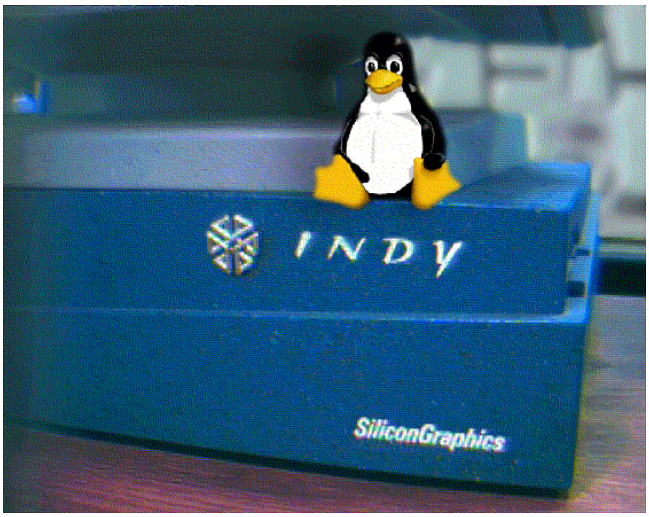
```
Processor, Processes -- times in microseconds -- smaller is better
----------------------------------------------------------------
Host                 OS  Mhz null null      open selct sig  sig  fork exec sh
                             call  I/O stat clos       inst hndl proc proc proc
--------- ------------- ---- ---- ---- ---- ---- ----- ---- ---- ---- ---- ----
mips-linu  Linux 2.1.99  100  1.6  3.1   39   43 0.21K  6.3   24 2.9K  34K 118K
mips-linu  Linux 2.1.99  100  1.6  3.1   38   42 0.21K  6.3   24 2.8K  34K 118K
mips-sgi-      IRIX 6.2   99  5.2  16.  395  496 0.76K 16.7   97 5.9K  19K  56K
mips-sgi-      IRIX 6.2   99  5.2  16.  406  509 0.41K 16.7   97 5.9K  19K  56K
```

Výpis č. 4: lmbench 2beta6

SGI/Linux — projekt ve stavu alfa 1, skoro 2

Jan Pazdziora, 10. července 1998

Port Linuxu na stanice Indy a Challenge S firmy Silicon Graphics se dostal do již vcelku zajímavého stadia. Kromě jádra, které se chová velmi stabilně, jsou již na architekturu mipseb (MIPS, big endian) přeloženy více než čtyři stovky RPM balíků z distribuce Red Hat 5.1, navíc je v jisté podobě portována i instalační část, takže po nastavení bootp a NFS serverů vše probíhá již na modré obrazovce s menítky. Podporované jsou procesory R4400 až R5000, ale situace se mění každým týdnem. Nejdůležitější část, která stále chybí, je X server — ani XFree86 ani Xsgi ještě na SGI/Linuxu nechodí, takže pokud potřebujete na konzole pracovní stanice pracovat, asi ještě chvíli zůstanete u IRIXu. Na druhou stranu je toto vedle vývoje jádra nyní hlavní cíl, takže ta chvíle snad nebude tak dlouhá.



V každém případě můžete zkusit Linux na Indy nainstalovat — až bude X server, budete mít před ostatními náskok. Případně můžete k lepší budoucnosti přispět aspoň dumpem registrů či jinými zajímavými hláškami. Linuxová konzola vypadá na SGI opravdu pozitivně.

SGI/Linux jsem instaloval na stroji Indy s procesorem R4600PC, port Manhattanu označený Alpha 1. Podrobný popis instalace najdete spolu s dalšími informacemi o projektu na adrese(1). Je nutné mít již na disku vytvořené partitiony, já jsem za tímto účelem připojil starý externí 300MB disk, a použil IRIXový `fx`. Distribuce softwaru je primárně na serveru (2), na síti TEN-34 CZ je denní mirror na (3). Instalace Manhattanu má zagzipovaná kolem 170 MB (je to soubor `installfs.tgz`), takže je dobré mít rychlou síť.

Potřebujeme nastavit bootp, tftp a NFS tak, abychom bootovali soubor `vmlinux` a brali správný nfsroot. Potom v Command promptu dáme

```
boot bootp():/vmlinux
```

a dále by již všechno mělo být povědomé.

V Alpha 1 instalaci je problém s vytvořením swapu, je nutno ignorovat a swap posléze vytvořit `mkswapem` ručně. Dále některé RPM balíky hlásí, že nebyly nainstalovány korektně, i když byly. Také ignorovat. Jistá zkušenost a nadhled se vyplatí — instalaci jsem vinou starého disku, který se odmlčuje a občas ani není nalezen při bootu, dělal několikrát a podařilo se mi dokonce vytvořit systém, kde úplně chyběly soubory `/etc/passwd` a `shadow`.

Nainstalovaný Linux pak nejlépe nabootujeme tak, že umístíme linuxové jádro na IRIXovou partition a pustíme

```
boot vmlinux root=/dev/sdb1
```

či jiné odpovídající jméno linuxové partition. Kýžená odpověď zní

```
Red Hat Linux release 5.1 (Manhattan)
Kernel 2.1.99 on a~mips
```

Jádro 2.1.99 se občas chová trošku podivně — hodnota bufferů je velmi vysoká a práce systému se zpomaluje. Pomůže spustit proces, který naalokuje hodně paměti, po jeho skončení je systém zase živý.

Podle zpráv z SGI/Linuxového mailing listu by Alpha 2 verze portu Red Hatu 5.1 měla být hotova v době, kdy čtete tyto řádky, a má být zase o 1000 procent lepší. Bude už umět všechny čtyři instalační metody, nejen přes NFS, bude mít samozřejmě odstraněny všechny dosud nalezené chyby a bude obsahovat zase více dodělaných RPM balíků.

Abych ukázal, že Linux chodí velmi obstojně, zde je kus výstupu z `lmbench 2beta6` (verzi 1 se mi kvůli problémům kompilátoru nepodařilo kompletně přeložit).

Pokud máte někde poblíž Indy stanici, která nemusí běžet stále, máte navíc volný disk, rozumné připojení třeba na TEN-34 CZ a dvě tři hodinky času, které jste ochotni věnovat stažení instalace, nastavení bootovacího prostředí, instalaci s malými muškami a následně nadšení (či zklamání `;-)`) z Linuxu na SGI, rozhodně to zkuste. Především na školách by se nějaké Indy stanice mohly najít. Pokud jste již zpychli a systém bez X serveru a bez podpory audia nechcete ani vidět, nezbývá než ještě nějakou dobu počkat.

Adresu mailing listu, kam můžete poslat zprávu o výsledku, najdete spolu s jeho archivem na výše uvedeném WWW serveru. ■

1 WWW server projektu SGI/Linux
http://www.linux.sgi.com

2 FTP server projektu SGI/Linux
ftp://ftp.linux.sgi.com
3 Mirror na síti TEN-34 CZ
ftp://ftp.fi.muni.cz/pub/linux/sgi

Caldera Netware Server 4.10 Beta (část 1)

Milan Šorm, 8. července 1998

Už několik let sleduji hodně podrobně pokusy vytvořit více či méně dobrý emulátor služeb serveru Novell NetWare pro Linux. Doposud jsem vyzkoušel český `lwared` i německý `mars-nwe`, které dokáží poměrně obstojně emulovat bindery databázi a služby NetWaru 3.1x. Nedávno (resp. v červnu) uvolnila Caldera beta verzi svého Caldera NetWare Serveru (CNS), který umí NDS, print server a má být po uživatelské stránce plně kompatibilní s NetWarem 4.10. Za tímto účelem Caldera zaplatila Novellu licence na NDS, jádro NetWaru 4 apod.

Volně šiřitelnou variantu CNS pro tři uživatele si může každý sám nainstalovat jak do Caldera OpenLinuxu, tak i do jakékoliv jiné distribuce Linuxu. V tom druhém případě však můžete čekat menší či větší problémy. Současná betaverze je provozovatelná až do 1. října t.r., kdy jí vyprší platnost. To už by ale dávno (Caldera slibuje 24. červenec) měla existovat ostrá verze. Více licencí než 3 si prý bude možno dokupovat od Caldery na CD v obvyklých baleních po 5, 10, 25 a více licencích. O cenách Caldera zatím nepíše nic.

Já jsem se rozhodl nainstalovat CNS na svůj RedHat 5.0 na Pentiu Pro s 64 MB paměti, abych dosáhl odpovídající rychlosti jako s originálním NW serverem (a mohl srovnávat). Původně jsem hodlal použít jádro 2.0.35pre3, ale na stránkách Caldery jsem byl ujištěn, že budu muset jádro opatchovat a patche jsou maximálně pro 2.0.34.

Můj záměr ponechat pro Calderu jen 60 MB se záhy vymstil a bylo nutné zhruba 100 MB (včetně místa pro překlad jádra). A během instalace také neprobíhalo nic hladce, protože jsem si nejprve nevšiml dodávaného souboru `INSTALL` a zkoušel to rozchodit naslepo. To se mi silně nevyplatilo (nijak jsem např. nepoznal, které patche vlastně potřebuji). Později jsem sáhl po výše uvedeném souboru, ale stále to nebyla procházka růžovým sadem.

První fází bylo přizpůsobit systém (RedHat) Caldeře, tzn. nainstalovat některá RPMka a provést několik zásahů. Tzn. nainstalovat balíček `lisa-3.0-8.i386.rpm`, `SysVinit-scripts-1.02-5.i386.rpm` a `ipx-1.0-8.i386.rpm`, všechno samozřejmě s `--nodeps --noscripts --force`, protože závislosti v Caldeře jsou značně odlišné oproti RedHatu. Všechny balíčky se dají sehnat na (1).

Další fází bylo nainstalovat RPM balíček `ncurses-4.1-1.i386.rpm`, bez kterého se server odmítl instalovat. To však šlo až tehdy, když jsem zřídil symbolický link `/usr/share/termcap` ukazující na `/usr/lib/termcap`. Pak jsem musel zpětně upravit věci v `/etc/inittab`, `/etc/rc.d/` a `/etc/sysconfig/`, protože nyní odmítaly naopak pracovat originální RedHatí věci. Nebylo to zase tak složité. Podstatný rozdíl byl ten, že balíčkům Caldery se v RedHatu musí na začátek jejich `*.init` skriptů vložit řádek `SUBSYS=název`, samozřejmě místo `název` se napíše název skriptu.

Nyní se dá přikročit k nainstalování balíčků s vlastním serverem, tedy `netware-4.10b-20.i386.rpm`, `netware-sysvol-4.10b-15.i386.rpm` a RPM balík `pserver-4.10.1-5.i386.rpm`. To jde celkem bez problémů, pokud se vám podařilo připodobnit RedHat ke Caldeře pomocí předchozích kroků. Tato část instalace vás vyzve k modifikaci souboru `/usr/src/linux/include/sem.h`, kde je nutné změnit konstantu SEMMSL z původních 32 na 256. Do konfigurace překladu jádra je nutné zaznamenat podpora pro IPX protokol, která ovšem může být z modulů (tj. `<M>` The IPX Protocol). Většina souborů CNS je umístěna (po instalaci) v adresáři `/opt/netware`, `/var/opt/netware` a `/etc/opt/netware`.

Soubor `/etc/rc.d/init.d/netware`, což je hlavní spouštěcí skript pro CNS, je nutné rozšířit o řádky `SUBSYS=netware`, `PROBABLY=goodfin`, `NETWORKING=yes` a `SVIlock=/var/lock/subsys/netware`. Podobné změny je třeba udělat i v souborech `ipx` a `ipxripd` tamtéž. Opět jde o přizpůsobení RedHatu Caldeře.

Nyní přichází na řadu opatchování jádra, které zase neproběhne tak hladce, jak člověk očekává. Vychází se z čistého jádra 2.0.34 (např. ze serveru (2) nebo od Caldery). Nejprve se aplikuje `ipx-patch`, který se najde na Caldera FTP site a jmenuje se `linux-2.0.34-ipx.patch` (obvyklým postupem `patch -p0 <linux-*-ipx.patch` v adresáři `/usr/src`). Potom se někam mimo jádro rozbalí soubor `streams-1.30.98.tar.gz`, ve kterém se spustí `./install.sh 33` (protože streams umí jen jádra po 2.0.33, avšak jede i s 2.0.34 jak jsem si podle návodu vyzkoušel). Tento skriptík aplikuje patch do jádra, neudělá však všechno potřebné. Dalším krokem je nastavit pro překlad jádra podporu Streams (úplně poslední položka menu) a to podporu do modulů (`<M>`). Po provedení `make dep ;make clean` je potom nutné přejít do adresáře `drivers/streams/LiS/util/linux` a přeložit podpůrné programy příkazem `make all install`. Tyto programy se nainstalují do `/usr/sbin`. Teď už je možné jádro přeložit, přeložit moduly a obojí nainstalovat obvyklým způsobem. Do výsledného adresáře `/lib/modules/2.0.34/misc` pak přidáme i moduly z `/opt/netware/lib/*.o` — moduly užívané CNS, ale nejsou k nim zdrojové texty.

Teď je nutné udělat ze systému anglický systém podobný Caldeře, aby byl CNS schopen zjistit expiration time. To se udělá nastavením `LC_ALL=C` před spuštěním `nwserver` v `/etc/rc.d/init.d/netware` a uděláním symbolického linku z `/usr/share/zoneinfo` na `/usr/lib/zoneinfo`. Musí také existovat `locale/C` v locales, protože jinak se přes hlášení o expiraci nedostanete.

V README streamů je popsán ještě způsob jejich otestování (zajímavé, ale značně nesrozumitelné) a doporučení umístit do `/etc/rc.d/rc.local` řádek `cat /dev/runq &`. Jak jsem však zjistil, toto stejně neslouží k ničemu jinému než k vypisování nesmyslů (zdánlivých) na konzolu. Podstatně důležitější je vytvoření dalších speciálních souborů do `/dev`, což se udělá přeložením `make makenodes` v `drivers/streams/LiS/head/linux` a následným `./makenodes` tamtéž.

Dále je nutné nastavit IPX subsystém v souboru `/etc/sysconfig/ipx` podle souboru `INSTALL`, jde hlavně o číslo IPX sítě apod. Podobně je nutné ještě nastavit parametry v `/etc/sysconfig/daemons/ipx` a `ipxripd` a `nwclient` tamtéž. Do souboru `/etc/opt/netware/nwconfig` se zapíše jméno serveru.

Než provedeme reboot je ještě vhodné vytvořit

do /opt/netware/lib symbolický link ke knihovně libc.so.5. Potom už můžeme provést reset a sledovat výpisy na obrazovce. Chybové výpisy CNS se objevují na VT10 (přístupná přes Alt F10) a v souboru /var/log/netware/ncpslog.

A tady narážím na problém, že mi běží sapd démon i routequeue, ale nwserver mi běžet nehodlá. Vypisuje mi hlášení o Virtual process identifier table a nepracuje. Uvidíme, co mi na toto hlášení poradí v konferenci o CNS a v příštím čísle Linuxových novin se s vámi podělím o zkušenosti z provozu. ■

1 Caldera FTP server
ftp://ftp.caldera.com
2 Zdrojové texty Linuxu
ftp://ftp.kernel.org

Pracovní soubory na ramdisku

Milan Šorm, 8. července 1998

Donedávna jsem byl zvyklý pracovat s Linuxem na slabších strojích s několika málo megabyty paměti. Výkonnější stroje byly vždycky přisouzeny na podivné operační systémy a Linux byl degradován na operační systém starých a slabých strojů.

Bohudík se tento názor začal vytrácet, a tak jsem byl postaven před stroj s 64 MB paměti, který měl dělat WWW server, obsluhovat poštu a příležitostně na něm měly běžet překlady. A právě ty mě podnítily zabývat se myšlenkou, proč se nevrátit do éry MS-DOSu, kdy měl každý svůj malý ramdisk a na něm překládal knihovny a své programy. Řada věcí se tím urychlila.

Proto jsme se s kamarádem pustili do realizace tohoto plánu. Chtěli jsme 32 MB ponechat beze změny, avšak ze zbylých 32 MB udělat ramdisk, na něj umístit /tmp a v případě potřeby do tohoto adresáře nalinkovat ještě jiné adresáře (např. se zdrojáky).

Nejprve bylo nutné přeložit jádro s podporou ramdisku. To provede nastavení volby CONFIG_BLK_DEV_RAM=y v souboru .config. Jádro se přeloží a zavede se běžným způsobem do lilo.conf. Navíc mu však přibude volba ramdisk=32768, která udává velikost interního členění ramdisků v jádře. Po nabootování tohoto jádra lze potom využít až 16 ramdisků o této velikosti (my použijeme jen jediný). Tyto ramdisky jsou přístupné přes /dev/ram0 až /dev/ram15.

My jsme chtěli mít ramdisk zakládaný běžným init-skriptem známým v RedHatu a tak jsme do /etc/rc.d/init.d udělali skript nazvaný ramdisk, který reagoval na povely start založením ramdisku, namountováním na /tmp a nakopírováním nějakých základních dat z /usr/local/tmp. Na povel stop potom ramdisk odpojil. Při tvorbě tohoto skriptu jsme vyšli z podobných skriptů (např. pro sshd).

Příslušné povely pro vytvoření ramdisku jsou mkfs.ext2 -b 1024 /dev/ram0 32748, který založí ext2 filesystém na prvním interním ramdisku. Tento filesystém bude mít 32748 kilobytových bloků. Ne všechny budou využitelné pro data, ale většina ano (část zabere superblok, tabulky inodů apod.). Připojení filesystému k /tmp provede mount -t ext2 /dev/ram0 /tmp a dále je nutné nastavit t-bit adresáři /tmp, což učiní povel chmod 1777 /tmp. No a na závěr už stačí jen nakopírovat nějaká implicitní data povelem cp -R /usr/local/tmp /.

Odpojení ramdisku v sekci stop zvládne jistě každý sám. Stejně tak zajistit spouštění tohoto skriptu v příslušných běhových úrovních (3,5 apod.). A proč kopírujeme implicitní data? Máme některé veřejné uživatele, kteří mají jako svůj domovský adresář nastaveno právě /tmp a jako shell nějakou konkrétní informační službu. A některé takové služby potřebují konfigurační soubor, který tam nakopírujeme tímto způsobem. Navíc máme na /tmp napojené i jiné adresáře, ve kterých vznikají dočasná data proměnlivé velikosti.

Ramdisk lze užít i jiným způsobem. Lze si například vyrobit několik menších ramdisků a vytvořit si skript, který uživateli na požádání připojí takovýto ramdisk do některého jeho podadresáře v home. Takovýto uživatel si na něm něco může soukromě překládat a potom zase zrušit připojení (příp. se o to může např. večer postarat nějaký skript spouštěný z cronu). Fantazii se meze nekladou... ■

Chyby uživatelů patří uživatelům

Milan Šorm, 9. července 1998

Umožnil jsem tak jako mnohý správce uživatelům vytvářet vlastní webové stránky a zveřejňovat je běžnou konvencí. Rovněž jsem jim dovolil psát si vlastní CGI skripty. A to s sebou nese jeden zásadní problém.

Chyby CGI skriptů jsou standardně zapisovány do chybového výstupu Apache, resp. celý chybový výstup (STDERR) je připojen do tohoto souboru s identifikací skriptu a času vzniku problému.

Jenže chyba v uživatelském skriptu zajímá především uživatele, který skript vytváří. Nehledě k tomu, že se potom na partition, na které se ukládají systémové logy, objevují záznamy uživatelů a oni mají teoreticky možnost tuto partition zahltit.

Existuje více řešení tohoto problému. Základním řešením, které řeší informovanost uživatele i zatěžování centrální logů, je nesystémové užívání přesměrování chybového vstupu někam jinam samotným tvůrcem CGI skriptu. Např. v Perlu se naskýtají pěkné možnosti při užití CGI::Carp či Tie::STDERR|. Proč je toto řešení nesystémové? Protože ponecháváme na vůli tvůrce, jestli toto využívá.

Jiné řešení, které řeší jen informovanost uživatelů, spočívá v jednorázové kontrole chybového logu Apache např. večer. Nalezené chyby uživatelů jsou jim potom nějakým způsobem předány (zaslány mailem, připojeny do souboru chyb apod.). Ani toto řešení není ideální, i když jej lze kombinovat s promazáváním logu příp. zkrácením doby kontroly např. na jedinou minutu.

Za nejlepší variantu považuji možnost nasadit na chybový výstup Apache průběžný filtr. Já nejsem Apachem nucen psán všechny chyby do souboru, ale mohu si je nechat posílat do filtru, který bude tyto chyby procházet a rozesílat příslušným uživatelům či zapisovat do systémového logu.

Jak se to provede? Do řádku ErrorLog v souboru httpd.conf se místo jména souboru uvede |/usr/sbin/analerrorlog. To je samozřejmě kolonová konvence (trubková zní lépe ;-) následovaná jménem filtru. Filtr bude spuštěn jen jedenkrát (při startu Apache) a po celou dobu běhu jsou mu na standardní vstup předávány údaje, které by chodily do error_logu.

Toto je např. standardní jednořádkový chybový výpis při neexistenci stránky:

```
[Wed Jul 8 18:48:20 1998] access to\
/www/htdocs/plan.htm failed for \
infra.euroseek.net, reason: File does not exist
```

Toto se uloží v hlavním `error_logu`, kterým se probírá náš webmaster.

A teď komplexnější ukázka z uživatelských CGI skriptů, když nastane chyba:

```
Illegal division by zero at\
 /home/popelnik/html/perl/passwd.pl line 35.
[Tue Jul 7 07:47:53 1998] access to\
 /~popelnik/passwd.cgi failed for \
mensa.physics.muni.cz, reason:\
 Premature end of script headers
```

Toto se uloží k uživateli do jeho `error_log` souboru.

Bohužel řádek, který označuje, ve kterém skriptu k chybě došlo, přijde vždy až jako poslední. Proto je nutné si zapamatovat všechny procházející řádky až po řádek začínající hranatou závorkou (např. v souboru). Z hranaté závorky lze již snadno rozpoznat, komu skript patří (za znakem ~ následuje login uživatele) a je možné mu příslušná data zaslat (mailem, uložením do nějakého souboru v jeho home adresáři apod.). Pokud se nenajde uživatel, komu by chyba patřila, může se uložit do původního chybového souboru.

Filtr navíc může převádět Redirect údaje a Alias údaje zpět na původní majitele (pokud máte např. konvenci s plnými jmény uživatelů namísto s loginy) či ignorovat některé chyby (typicky věci obsahující řetězce „send body“, „lingering close“, „lost connection“, „connection reset by peer“, „fixed spelling“ či jména robotů — vyhledávačů.

Vhodným jazykem pro napsání filtru je např. Perl. Průběžně lze chyby místo v souboru zachycovat v paměti (pole) a na závěr vyklopit např. přes `sendmail` uživateli do pošty. A to se potom chybující uživatel nad svým CGI skriptem rychle zamyslí.

Samozřejmě lze stejný způsob aplikovat i na soubor přístupů (`access_log`) a zaznamenávat si transakce, ale to lepší řešit dávkově a data dávkově přenášet do grafů. Alespoň si to tak myslím já. ■

Hlasové modemy a jejich použití

Jan Kasprzak, 17. července 1998

S modemem jste se jistě již setkali. Asynchronní modem je zařízení, které je schopno přenášet data po telefonní lince nebo po metalickém okruhu (analogová pevná linka). Jistě jste si také všimli, že některé modemy mají někde u své specifikace připsané slovo *Voice*. A právě o těchto modemech je článek, který právě čtete.

Co je hlasový modem?

Jde o zcela běžný modem, určený pro komunikaci nad telefonní linkou. Má rozhraní pro připojení k telefonní síti a rozhraní pro připojení k počítači, obvykle RS–232. Hlasový modem má oproti běžnému modemu schopnost přehrávat do telefonní linky digitální zvukový signál, který dostane z počítače, a naopak počítači předávat digitalizovaný zvuk, který „slyší“ v telefonní lince. Většina hlasových modemů má také schopnost tento zvuk do jisté míry analyzovat — například detekovat ticho na lince, vyzváněcí a obsazovací tóny, a také rozpoznávat DTMF (*Dual Tone Modulation Frequency*) tóny. To je to pípání, které vydává klávesnice všech novějších telefonních přístrojů při vytáčení čísla. Většina hlasových modemů má také vlastní reproduktor a konektor pro připojení sluchátek a mikrofonu.

K čemu je hlasový modem?

Asi nejzákladnější a nejjednodušší aplikací hlasového modemu je emulace telefonního záznamníku. Při příchozím volání software rozpozná, jde-li o datové (případně faxové) volání, nebo dovolal-li se člověk. V posledním jmenovaném případě pak do telefonu přehraje zprávu pro volající a umožní uložit hlasový vzkaz.

Pokud si ale uvědomíme, že pomocí kláves tónové volby může volající určitým poměrně dobře definovaným způsobem předávat hlasovému modemu informaci, zjistíme, že možnosti jsou daleko větší. Můžeme pak vytvářet i daleko rozsáhlejší aplikace. Jedna z možností je vylepšit náš telefonní záznamník tak, aby po zadání číselného „hesla“ pomocí tónové volby umožnil přehrát uložené zprávy nebo změnit zprávu pro volající. Takto se záznamníkem můžeme komunikovat i v případě, že nejsme u svého počítače. V dalším popíšu problém, který jsem řešil pomocí hlasových modemů (a Linuxu, samozřejmě), a dále uvedu prostředky, které jsem k řešení použil.

Přijímací zkoušky

Stejně jako u všech vysokých škol, konají se i na Fakultě informatiky přijímací zkoušky do prvního ročníku. Celé přijímací řízení končí tím, že uchazeč napíše test, ten někdo opraví (v případě FI ještě tentýž den), přičtou se body za střední školu, různé olympiády a podobně, uchazeči se setřídí podle dosaženého počtu bodů, v určitém místě se udělá čára oddělující přijaté a nepřijaté uchazeče, vytiskne se oznámení o výsledku řízení a rozešle se uchazečům (no, trochu jsem to zjednodušil, ale zhruba to takto probíhá). Uchazeč tedy má několikadenní dobu, po kterou čeká na výsledek zkoušky. Problém je ten, že ne každý uchazeč (nebo jeho rodiče) vydrží čekat až do písemného oznámení. Zaměstnankyně studijního oddělení pak po několik dní nedělají nic jiného, než zodpovídání dotazů na to, jak ten který budoucí génius `(:-)` dopadl.

Rozhodli jsme se, že studijnímu oddělení situaci trochu ulehčíme. Výsledky zveřejňujeme jednak přes WWW, a jednak pomocí hlasových modemů. Uchazeči už u přijímací zkoušky dostanou leták s informací, na jaké telefonní číslo mohou zavolat, a jakým způsobem se tam požadovanou informaci dovědí.

Rozhovor s počítačem

Celý systém je řešený tak, že uživatel zavolá na dané číslo, a uslyší toto:

> Dobrý den.
> Dovolali jste se na hlasový server Fakulty informatiky.
> Stiskněte tlačítko „0“ na svém telefonu.

Požadavek na stisk tlačítka nula slouží jednak k tomu,

abychom detekovali volající, jejichž telefon nepodporuje tónovou (DTMF) volbu, a jednak proto, že na dalších kódech pak můžeme v budoucnu zavést i jiné služby, než jen sdělování výsledků přijímacích zkoušek.

Stiskl-li uchazeč úspěšně tlačítko nula, systém pokračoval dále:

> Zadejte váš identifikační kód a stiskněte křížek.

Uchazeč zadal číselný identifikační kód a stiskl tlačítko [#]. Kódy se uchazeči dověděli při přijímací zkoušce. Kód sestával z šestimístného identifikačního čísla, použitého pro tohoto uchazeče i ve studijní databázi, a tří náhodně vygenerovaných číslic, které sloužily jednak jako heslo, a jednak měly snížit možnost toho, že někdo omylem vyslechne výsledek někoho jiného a vyvodí z toho pro sebe nesprávné závěry. Tlačítko [#] sloužilo jako ukončovací znak pro sekvenci číslic. Tento způsob zadávání čísla zamezí tomu, že uživatel a systém budou na sebe navzájem čekat, pokud uživatel stiskne o jedno tlačítko méně nebo systém některý tón nerozpozná.

Pokud uchazeč zadal chybný kód (například 1234#, systém odpověděl:

> Neplatný kód číslo jedna dva tři čtyři.
> Zadejte váš identifikační …

Zadal-li uživatel chybný kód (nebo nezadal-li kód) třikrát, systém ukončil spojení. Pokud zadal platný kód, systém jej přečetl a oznamoval výsledky. Několik příkladů zde uvádím:

> Uchazeč s identifikačním číslem jedna dva tři čtyři se nedostavil k přijímací zkoušce.
> Uchazeč s identifikačním číslem jedna dva tři čtyři získal sto devět bodů za písemnou přijímací zkoušku, pět bodů za střední školu, celkem sto čtrnáct bodů, a tedy nebyl přijat ke studiu. K přijetí bylo potřeba aspoň sto sedmdesát bodů.
> Uchazeč s identifikačním číslem jedna dva tři čtyři získal devadesát bodů za obor výpočetní technika, sto dvacet tři bodů za obor fyzika, patnáct bodů za střední školu, celkem dvěstě dvacet osm bodů, a tedy byl přijat ke studiu. K přijetí bylo potřeba aspoň dvěstě deset bodů.

První příklad ukazuje někoho, kdo ani ke zkoušce nepřišel, druhý je uchazeč na odborné studium, který nebyl přijat, a třetí je uchazeč na učitelství výpočetní technika–fyzika, který byl přijat.

Na konci rozhovoru se systém rozloučil a zavěsil:

> Děkujeme za zavolání.

Později jsme systém ještě modifikovali pro použití k přijímacím zkouškám na Ekonomicko-správní fakultu MU. Tam systém navíc říkal iniciály jmen uchazečů, obory, na které byli přijati a pořadí, na kterém se umístili. Na druhé straně ESF nepožadovala zabezpečení heslem, takže jako vstupní kód sloužilo jen osobní číslo uchazeče.

Použitý hardware

V době prvních testů asi před půl rokem jsme používali počítač s procesorem i486/66, 16MB paměti a asi 250 MB disk. V době konání zkoušek jsme už měli Pentium 90 MHz a 32 MB RAM. Na tomtéž počítači ale neběžely jen hlasové modemy — slouží jako tiskový server, faxový server, jsou na něj přes sériové porty napojeny konzoly nejrůznějších zařízení a vykonává ještě několik dalších služeb. Moje zkušenost je, že na zátěži procesoru této třídy se vůbec neprojevilo, jestli běželo hlasové spojení, nebo ne. A jak uvidíme dále, s efektivitou obslužných programů jsme se nijak neobtěžovali — značná část systému je psána v Perlu. Na kvalitě hlasového spojení se také nijak neprojevila případná zátěž hlasového serveru.

V počítači jsou dvě osmiportové sériové karty Cyclades Cyclom 8Yo, každá v ceně zhruba 11 500 Kč. Moje zkušenosti s tímto hardwarem jsou ty nejlepší. Driver pro Linux vypadá stabilně a karty generují poměrně malou zátěž systému. Firma Cyclades dokonce uvádí Linux na prvním místě mezi podporovanými operačními systémy. Narazil jsem jen na jediný problém: driver pro tyto karty nefungoval, pokud byl kompilován překladačem `egcs`. Tohle se dost těžko detekuje v případě, že jste si `egcs` nainstalovali někdy před půl rokem a už ani nevíte, že na svém počítači nemáte `gcc`.

Pro hlasové aplikace jsme pořídili šest modemů. Měl jsem se rozhodnout mezi US Robotics Sportster a ZyXEL Omni (s Rockwell chipem). Nakonec to vyhrál Sportster — měl lepší zvuk a fungoval lépe. Jako zajímavost uvedu, že ke Sportsteru dostanete sluchátka, zatímco k ZyXELu Omni výrobce (nebo prodejce?) dává malinkou lahvičku J&B whisky…

Modemy jsme napojili na místní telefonní ústřednu, která se ukázala býti jediným slabším článkem systému. Ve špičce totiž vyřizovala až dva hovory za sekundu, což zřejmě bylo víc, než mohla unést. Toto bylo kritické zejména u zkoušek na FI, jejichž výsledky byly zveřejněny ještě tentýž den. A tedy kolem očekávané hodiny zveřejnění byl nápor celkem velký. U zkoušek na ESF situace nebyla tak kritická, protože zveřejnění bylo až skoro týden po vlastních zkouškách, takže se všichni uchazeči nesnažili volat skoro přesně v tutéž hodinu.

Propustnost systému je zhruba 150-200 hovorů za hodinu. Průměrný „rozhovor“ trval zhruba dvě minuty, což dává okolo 30 hovorů za hodinu na jeden modem.

Software

Jak se tedy pod Linuxem pracuje s hlasovým modemem? Možností je několik. Mně se nejvíc zalíbil program `vgetty`. Jde v podstatě o `mgetty` (1) s vestavěnými hlasovými funkcemi. Tento balík je distribuován v rámci `mgetty+sendfax` jako další dodatečná aplikace. Protože `mgetty` je špičkou ve svém oboru, lze očekávat, že ani `vgetty` na tom nebude špatně. Než přejdu k popisu toho, jak se s `vgetty` pracuje, uvedu jen, že zbytek systému byl Red Hat Linux 5.0 a jádro 2.0.34.

Program `vgetty` zajišťuje (téměř) jednotný přístup ke hlasovým modemům, takže odlišnosti mezi jednotlivými typy modemů schovává. Program razí myšlenku „jednoduché jednoduše, složité složitěji“ — sám obsahuje jednoduchý hlasový záznamník, a pro složitější problémy má podporu skriptů.

Hlasový shell

Komunikace vgetty s uživatelskou aplikací probíhá pomocí tzv. *hlasového shellu*. To je program, jehož jméno můžeme nastavit v konfiguračním souboru voice.conf (na mém Red Hatu byl v adresáři /etc/mgetty+sendfax/). V okamžiku příchozího volání spustí vgetty tento hlasový shell, předá mu v proměnných prostředí svoje PID (používá se pak pro komunikaci), a dále čísla dvou deskriptorů. Tyto deskriptory ukazují na dvě roury, které slouží k obousměrné komunikaci mezi vgetty a hlasovým shellem. Shell pak má možnost jednoduchým znakovým protokolem předávat vgetty svoje příkazy („přehrej tento soubor“, „pípni“, „nahrej zvuky, které slyšíš z linky, do tohoto souboru“, a tak podobně). Na druhou stranu vgetty předává shellu informace o stavu linky („Slyším DTMF tón #“, „Na lince je ticho“, „Slyším obsazovací tón, volající asi zavěsil“, atd.). Tento protokol umožňuje implementovat vše potřebné pro hlasové aplikace.

Perl na scéně

Program vgetty obsahuje několik příkladů hlasových shellů — vesměs jde o skripty v Bourne Shellu. Autor píše, že možná někdy bude podporovat i Perl. Protože v Perlu se přece jen píše lépe než v shellu, rozhodl jsem se, že přechod na Perl trochu urychlím. V době vydání tohoto čísla Linuxových novin by již měl být na CPANu (2) dostupný modul Modem::Vgetty, který je výsledkem mé práce. Hlavní kód je již stabilní, momentálně dokončuji drobné detaily jako je například dokumentace.

Zmiňovaný modul je objektově orientované, událostně řízené rozhraní k vgetty. Nejjednodušší aplikaci, která přehraje zvukový soubor volajícímu, můžeme napsat třeba takto:

```
use Modem::Vgetty;
my $v = new Modem::Vgetty;
$v->play_and_wait('/cesta/soubor.rmd');
exit 0;
```

Je vidět, že se zde používá jakýsi soubor s koncovkou .rmd. Znamená to *Raw Modem Data* a soubor obsahuje zvuková data přímo pro konkrétní typ modemu. Toto je jediné místo, kde vgetty neskrývá před uživatelem typ modemu a kde aplikace musí tento typ znát. S programem vgetty se dodává i sada konverzních programů se souhrnným názvem pvftools — zavádí portabilní textový zvukový formát PVF (*Portable Voice Format*), přes který se provádí konverze na jiné formáty. Tento přístup trochu připomíná balík netpbm.

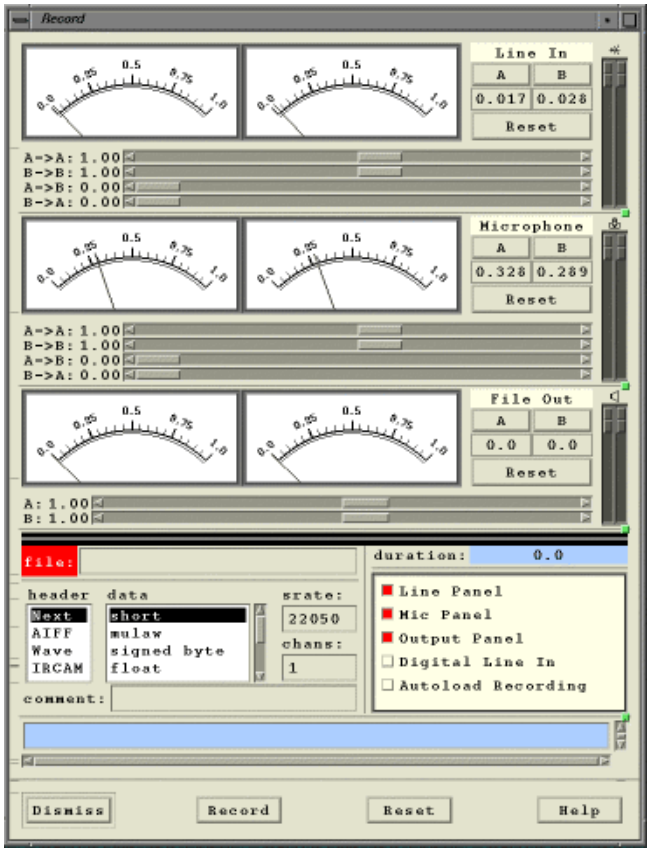
Dalším příkladem může být jednoduchý záznamník: volajícímu se přehraje zpráva a má možnost uložit svůj vzkaz.

```
use Modem::Vgetty;
my $v = new Modem::Vgetty;
$v->add_handler('BUSY_TONE',
'konec', sub { $v->stop; exit 0; });
$v->enable_events;
$v->play_and_wait('/cesta/welcome.rmd');
my $num = 0;
$num++ while(-r "/cesta/$num.rmd");
$v->record("/cesta/$num.rmd");
sleep 30;
$v->stop;
exit 0;
```

Zde už vidíme jednoduché použití událostí. Modul poskytuje několik druhů událostí. Jednou z nich je BUSY_TONE, která vzniká, detekuje-li modem na lince obsazovací tón. To je zejména v případě, kdy volající zavěsil. Program na tuto událost reaguje tak, že ukončí svoji činnost. Řetězec konec zde slouží jen jako identifikace handleru pro danou událost, pokud bychom časem tento handler chtěli zrušit.

Zvuková data

Pomocí modulu Modem::Vgetty jsem implementoval celý výše popisovaný hlasový systém. Vzhledem k tomu, že nemám uspokojivě fungující systém pro syntézu řeči, rozhodl jsem se, že celý text namluvím a rozdělím na jednotlivá slova. Myslel jsem si, že pro nasazení v praxi pak bude použit nějaký příjemný ženský hlas, ale nakonec tam zůstal ten můj.



Zvuky jsem vytvářel za pomocí mikrofonu, zvukové karty Gravis Ultrasound MAX a programu snd (3). Tento program umožňuje jak zaznamenávat zvuky, tak poměrně jednoduše tyto zvuky editovat. Tak například jsem do jednoho souboru namluvil všechny možné i nemožné číslovky, a pak jsem je pomocí tohoto programu ukládal po slovech do jednotlivých souborů. Je to nepříliš záživná práce, jak si jistě umíte představit, zejména u tak jednotvárných slov, jako jsou číslovky nebo jednotlivá písmena abecedy.

K číslovkám jsem si ještě napsal další modul (bude zřejmě uvolněn jako Cz::Speak nebo Cz::Numbers). Tento modul umožní převést zadané číslo na sekvenci českých slov. Tedy číslo 1234 převede na čtyři řetězce: „tisíc“ „dvěstě“ „třicet“ a „čtyři“. Je tam i několik rozšíření — například uživatel může specifikovat rod celého čísla (což se projeví

```
%changelog
* Thu Jun 11 1998 Jan "Yenya" Kasprzak <kas@fi.muni.cz>
- updated to 1.2.25.
- fixed truncated .pam patch.
- built on RedHat 5.1 (and tested upgrade on 5.0 too).

* Wed May 20 1998 Jan "Yenya" Kasprzak <kas@fi.muni.cz>
- Updated to 1.2.23:
- removed the -NYP patch (no longer necessary).
- removed the -tmpfile patch (no longer necessary).
- added %post scripts linking the ssh*1 binaries and manpages as ssh*
        if no such links exist. This will allow a coexistence of
        ssh1 and ssh2 RPMs on one system.
```

Výpis č. 5: Příklad sekce changelog

pouze pokud je číslo rovno jedné nebo dvěma), nebo přečíst číslo po trojicích cifer, po dvojicích cifer nebo i po jednotlivých cifrách. Také umí přečíst časový interval a některé další číselné údaje.

Spolupráce s databází

Vzhledem k tomu, že na projektu pracovalo více lidí (jednou z částí bylo například zveřejňování výsledků přes WWW), měli jsme jednotné rozhraní: databází byly DBM soubory a rozhraním byl program, který dostal jako parametr identifikační kód uchazeče a vrátil na výstupu informace o neplatném kódu nebo o uchazeči.

Závěr

Prokázalo se, že pro podobné účely je vgetty dostatečně mocný prostředek. Systém fungoval téměř bez výpadku a obsloužil řádově tisíc hovorů, při kterých byl zadán správný identifikační kód (což se podařilo volajícím zhruba v 70 procentech volání) v případě zkoušek na FI, a zhruba tři tisíce hovorů v případě ESF. U takovýchto větších akcí se vyplatí mít k dispozici náhradní stroj se stejným softwarem, aby v případě výpadku mohl být systém oživen pouhou výměnou počítače. Toto jsme naštěstí měli, a tak v okamžiku pádu hlasového serveru (dodnes uspokojivě nevysvětleno), kdy došlo k poškození kořenového adresáře a ztrátě více než poloviny souborů na disku, jsme mohli během zhruba hodiny zapojit náhradní počítač a věnovat se restaurování původního počítače ze zálohy (také pravidelně zálohujete, není-liž pravda :-). Na druhé straně tato technologie má své omezení. Asi největším z nich je nutnost manuálního namlouvání hlasových dat do počítače. V případě rozsáhlejších systémů bychom pak narazili na problém, že případné úpravy by musel mluvit tentýž člověk. Kdybychom měli rozumně fungující syntézu řeči, byla by situace trochu jiná. ■

1 Mgetty
ftp://ftp.leo.org/pub/comp/os/unix/networking/mgetty/
2 CPAN
ftp://ftp.fi.muni.cz/pub/perl/
3 Program snd
http://www-ccrma.stanford.edu/CCRMA/Software/snd/snd.html

Další sekce spec-souboru

Jan Kasprzak, 17. července 1998

V minulém čísle našeho seriálu o systému RPM jsem začal detailně popisovat formát spec-souboru, který řídí vytváření RPM balíku. Vysvětlil jsem, jaký význam mají jednotlivé tagy v hlavičce souboru, a nyní budu pokračovat výkladem o dalších sekcích spec-souboru. Některé sekce jsou povinné, jiné nepovinné. Sekce začíná znakem % a svým názvem na začátku řádku. Pořadí sekcí ve spec-souboru není definováno — zde budou jednotlivé sekce uvedeny v jakémsi logickém uspořádání.

Se sekcí %description jsme se už setkali. Tato sekce je povinná a obsahuje textovou informaci o určení a použití balíku. Tuto sekci využívá například instalační program Red Hat Linuxu, když se klávesou F1 dotážeme na podrobnější informace o balíku.

Další sekce je nazvaná %changelog a jak již její název praví, zachycuje protokol o změnách ve spec-souboru (nikoli tedy v softwaru samotném. Jednotlivé záznamy začínají hvězdičkou na první pozici řádku, následuje datum a jméno člověka, který změnu provedl. Na dalších řádcích pak může být popis změny. Příklad části sekce %changelog z balíku ssh je na výpisu Příklad sekce changelog.

Sekce %prep

Tato sekce již není jen pouhý text, ale je to shellový skript. Úkolem tohoto skriptu je rozbalit příslušné zdrojové archivy a nainstalovat potřebné záplaty. Tento skript (stejně jako skripty z dalších sekcí) je spuštěn shellem s parametrem -e, který říká, že shell má ukončit provádění skriptu s chybou, skončí-li libovolný z příkazů ve skriptu s chybou (s nenulovou návratovou hodnotou). Na tuto skutečnost je potřeba pamatovat při psaní skriptů. Není například možné používat konstrukce typu

```
test -r /etc/ssh/ssh_host_key && ssh-keygen blabla
```

Namísto toho je nutné použít následující ekvivalent:

```
if test -r /etc/ssh/ssh_host_key;
then ssh-keygen blabla; fi
```

Skript z této sekce, stejně jako ostatní skripty, má k dispozici některé speciální proměnné prostředí. Uvádím je zde spolu s příklady hodnot nebo s implicitními hodnotami:

- RPM_SOURCE_DIR=/usr/src/redhat/SOURCES — adresář, ve kterém jsou uloženy zdrojové archivy a zá-

platy. Skript %prep odsud může rozbalovat tyto archivy a aplikovat záplaty.

- RPM_BUILD_DIR=/usr/src/redhat/BUILD — adresář, ze kterého je skript spuštěn. Sem se pak rozbalí jednotlivé zdrojové archivy a zde probíhá kompilace.
- RPM_DOC_DIR=/usr/doc — adresář, do kterého se ukládá dokumentace k balíku.
- RPM_ARCH=i386 — architektura, pro kterou se aktuální balík vytváří.
- RPM_OS=Linux — operační systém, pro který se aktuální balík vytváří.
- RPM_OPT_FLAGS="-O2 -fno-strength-reduce -m486" — optimalizační přepínače pro kompilátor jazyka C.
- RPM_BUILD_ROOT=/tmp/ssh-root — specifikace adresáře BuildRoot.
- RPM_PACKAGE_NAME=ssh — jméno balíku.
- RPM_PACKAGE_VERSION=1.2.25 — verze softwaru.
- RPM_PACKAGE_RELEASE=1i — verze RPM balíku.

Makra

Skript %prep vykonává ve většině balíků podobnou činnost — vezme soubor SourceN a rozbalí jej v příslušném adresáři, měl by nastavit správně vlastníka a přístupová práva vůbec, a dále aplikuje záplaty. Tato činnost se opakuje natolik často, že pro ni systém RPM poskytuje *makra* — tedy zkratky pro více příkazů. Takový typický %prep-skript s jedním zdrojovým archivem a jednou záplatou pak vypadá asi takto:

```
%prep
%setup
%patch -p1
```

Makro %setup vezme zdrojový archiv, uvedený v hlavičce jako Source, resp. Source0, rozbalí jej do adresáře BUILD, změní vlastníka všech souborů na superuživatele (to pouze v případě, že balík vytváří superuživatel) a nastaví bity r a x pro všechny uživatele.

Makro %patch aplikuje záplatu, uvedenou v hlavičce jako patch, resp. patch0. Přepínač -p1 z našeho příkladu je předán programu patch(1).

Skript %prep samozřejmě nemusí sestávat pouze z volání maker %setup a %patch. Může obsahovat i další příkazy, například je-li program konfigurován pomocí systému GNU autoconf a modifikujeme-li konfigurační soubor configure.in, můžeme v této sekci přegenerovat skript configure příkazem autoreconf (tuto konstrukci naleznete mimo jiné i ve spec-souboru Secure Shellu — tam se přidává podpora pro PAM do configure.in, a musí se tedy přegenerovat configure. Kromě toho makra mohou mít i parametry, kterými se specifikují další vlastnosti. K těmto vlastnostem se vrátíme v dalším díle našeho seriálu.

Sekce %build — kompilace softwaru

Tato sekce, stejně jako předchozí, je interpretována jako shellovský skript. Tento skript by měl provést vlastní kompilaci softwaru. Ve většině případů si vystačíme s následující formou:

```
%build
./configure
make CFLAGS="$RPM_OPT_FLAGS" all
```

Příkazu configure, pokud jej program podporuje, můžeme samozřejmě dávat i další parametry (třeba --with-gcc --prefix=/usr). Tohle je vůbec dost diskutabilní věc: podle Linux Filesystem Hierarchy Standard (FHS, dříve FSSTND) by lokálně instalovaný software měl být instalován do adresáře /usr/local, zatímco software, který patří k systému samotnému by měl být v /usr. Ovšem je software, který zabalíte do RPM a distribuujete, součástí systému, nebo je spíš blíže lokální instalaci? Můj názor na tuto problematiku je ten, že /usr/local bylo zavedeno proto, aby bylo možné ty lokálně instalované věci odlišit. Ale máme-li RPM, máme na odlišení jiné a lepší prostředky (například položku Distribution nebo Packager v hlavičce RPM balíku). Moje RPM balíky se tedy instalují přímo do systémových adresářů, nikoliv do /usr/local.

Sekce %install — instalace softwaru

Instalační sekce je také shellovský skript. Účelem tohoto skriptu je přemístit konfigurační soubory, zkompilované binární soubory, dokumentaci a různé další soubory na místo jejich použití. Úplně nejjednodušší formou této sekce je příkaz make install. Ale chceme-li umožnit rekompilaci RPM balíku i uživateli, který není superuživatelem, měli bychom spec-soubor navrhovat tak, aby používal tag BuildRoot. Pak instalace probíhá do adresáře $RPM_BUILD_ROOT místo kořenového adresáře. Většina Makefile-souborů předpokládá, že bude k dispozici aspoň základní struktura adresářů a souborů v systému. Proto většinou v této sekci musíme potřebný strom adresářů předem vytvořit. Sekce pak může vypadat takto:

```
%install
mkdir -p ${RPM_BUILD_ROOT}/usr/{bin,lib,man/man1}
make prefix="${RPM_BUILD_ROOT}/usr" install
```

Je dobré vědět, že ne všechny programy jsou schopny být kompilovány s jiným prefixem, než se kterým jsou pak instalovány. V takových případech nezbude než ručně zjistit, co přesně dělá příkaz make install, a jeho činnost pak nasimulovat v sekci %install s příslušným BuildRoot. U některých balíků je i tento postup příliš náročný, a proto se instalují přímo na místo určení. Takovýto balík pak ovšem může rekompilovat jen superuživatel. Příkladem takového balíku je například tetex.

Pokud balík nepoužívá BuildRoot, je podstatně náročnější jej odladit. Takovéto ladění a pokusná instalace by měly probíhat na jiném počítači, než vlastní kompilace. Může se totiž stát (a je těžké to odhalit), že do balíku zapomeneme přibalit některý důležitý soubor. Na původním počítači vše funguje, protože tento soubor tam zbyl po make install. Jinde balík ale selže.

Sekce %clean — smazání meziproduktů

Pokud při kompilaci vznikají nějaké meziprodukty mimo adresář BUILD, například v /tmp, je na sekci %clean, aby tyto meziprodukty promazala.

V příští části tohoto seriálu se dozvíme, jak specifikovat soubory patřící k balíku a jaké další skripty mohou ve spec-souboru být. ■

Zasmáli jsme se!

Pavel Janík ml., 17. července 1998

Nevím jak ostatní čtenáři Linuxových novin, ale já jsem se v červenci již jen smál, škola skončila, všechny zkoušky jsou jistě za námi a máme před sebou dlouhé tři měsíce nicnedělání a volna. No prostě fantazie. Ano, pouhá fantazie… Venku je zima, prší a člověk by chtěl něco dělat, ale v televizi je fotbal nebo se člověku prostě nechce. Ale to jistě všichni znáte.

V tomto čísle jsem si připravil tři zajímavé útržky, které dokázaly na mé jinak střízlivé tváři vyloudit úsměv.

Tibor Dancs si do své signatury doplnil následující řádek:

> Win95 … je jediné ovocie, ktoré padá
> v každom ročnom období.

Když jsem byl navštívit jednoho kolegu (Yenya říkal, že sem nemám psát jeho jméno, proto ho neuvedu) na jeho pracovišti, tak se mne zeptal, jak by mohl sejmout obrazovku instalačního programu. Poradil jsem mu, že by mohl vyzkoušet spustit instalační program na svém počítači. Co se stalo, můžete vidět na následujícím výpise:

```
$ /tmp/install2  --method ftp
you're running me on a live system!\
 that's incredibly stupid.
```

A nakonec jsem si nechal příspěvek v konferenci *net@cs.felk.cvut.cz*. Jeho autorem je Petr Bravenec, který se nenechal ovlivnit probíhající diskusí na téma cookies ano či ne a odpověděl i na jiný dotaz.

> Skripty jsou uloženy v cache, pokud jde
> o skripty volané metodou GET, tj. parametry
> jsou součástí URL. Pokud je použita metoda
> POST, browser se zeptá na REPOST.
> Řešením je buď použití GET nebo používání
> MS Internet Exploreru 3 — ten na metodu
> POST kašle úplně a skriptu nic nepošle.
> Nevýhodou použití MS IE3 je, že pokud si
> nutně potřebujete přečíst stránku, která
> „ještě teď fungovala“, může to zanechat stopy
> na vaší nervové rovnováze.

A to je v červencovém čísle Linuxových novin vše. Doufám, že se uvidíme zítra na Linux InstallFestu pořádaném sdružením CZLUG :-) ■

Linuxové noviny a jejich šíření

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

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

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

Registrované známky použité v tomto textu jsou majetkem jejich vlastníků.

Chtěli bychom poděkovat Fakultě informatiky Masarykovy university v Brně, INET, a.s., Juraji Bednárovi, Milanu Šormovi a společnosti OptiCom za poskytnutí diskového prostoru pro Linuxové noviny.

Linuxové noviny můžete najít na akademické síti TEN-34 CZ (3), na síti IBM Global Network na adrese (4), na serveru Gymnázia Vídeňská v Brně (5), na serveru časopisu Netáčik (6), který je připojen do slovenského SIXu, případně na serveru společnosti OptiCom (7).

Linuxové noviny jsou k dispozici také ve formátu HTML na adrese (8). ■

1 České sdružení uživatelů operačního systému Linux
http://www.linux.cz/czlug
2 Adresa redakce
mailto:noviny@linux.cz
3 Linuxové noviny na síti TEN 34-CZ
ftp://ftp.fi.muni.cz/pub/linux/local/noviny
4 Linuxové noviny na síti IBM Global Network
ftp://ftp.inet.cz/pub/People/Pavel.Janik/noviny
5 Linuxové noviny na komerční síti CESNET
http://www.gvid.cz/linux/noviny/
6 Slovenské zrcadlo Linuxových novin
ftp://netacik.sk/pub/linux/cz-noviny
7 Linuxové noviny — OptiCom
http://www.mathew.sk/noviny
8 Linuxové noviny ve formátu HTML
http://www.linux.cz/noviny

Šéfredaktor: Pavel Janík ml.
mailto:Pavel.Janik@linux.cz
sazba: Ondřej Koala Vácha
mailto:koala@informatics.muni.cz
jazykové korekce: Bohumil Chalupa
mailto:bochal@met.mff.cuni.cz
překlady: Hanuš Adler
mailto:had@pdas.cz
převod do HTML: Pavel Juran
mailto:xjuran@cs.felk.cvut.cz