Appendix A. The GNU General Public License

Table of Contents

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

59 Temple Place, Suite 330, Boston, MA 02111−1307 USA

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software−−to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

1. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

   Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.
2. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.
3. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   a. You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
   b. You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
   c. If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

   These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

4. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

    a. Accompany it with the complete corresponding machine−readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
    b. Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine−readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
    c. Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

   The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

   If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.
5. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.
6. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.
7. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.
8. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty−free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

   If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

   It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

   This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.
9. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
10. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

    Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.
11. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.
12. NO WARRANTY

    BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.
13. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year>  <name of author>

This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 2 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year  name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse–clicks or menu items––whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

Prev
Glossary

Home

Kapitola 1. Úvod do Slackware Linuxu

Obsah

Co je to Linux?

V počátku (r.1991) byl Linux osobním projektem Linuse Torvaldse. Hledal zůpsob, jak se dostat k Unixovému operačnímu systému a neutratit přitom moc peněz. A dále se chtěl učit vstupy a výstupy 386−kového procesoru. Linux byl uvolněn veřejnosti zdarma, takže kdokoliv ho mohl studovat a vylepšovat v rámci GNU General Public Licence (viz kapitolu *Open Source a Free Software* a v Příloze A).

Dnes Linux dorostl mezi hlavní hráče na trhu operačních systémů. Je portován na různé architektury jako je Compaq Alpha, Sun SPARC a UltraSPARC a Motorola PowerPC chipy (např. Apple Macintosh a IBM RS/6000). Linux je v současnosti vyvíjen stovkami (ne−li tisícovkami) programátorů z celého světa. Běží na něm programy jako Sendmail, Apache a BIND, které patří k nejpopulárnějším serverovým programům na internetu.

Pojem Linux ve skutečnosti představuje pouze kernel – jádro operačního systému. Tato část zodpovídá za dohled nad vaším procesorem, pamětí, harddisky a periferiemi. To je vše, co Linux dělá. Dohlíží nad operacemi ve vašem počítači a zajišťuje běh všech dalších programů. Všechny tyto další programy, které činí Linux užitečným, vyvíjejí nezávislé skupiny. Kernel a programy jsou potom různými společnostmi či jedinci dávány dohromady, čímž vzniká operační systém. Tomu pak říkáme "linuxová distribuce".

Kapitola 2. Help (nápověda)

Obsah

Můžou nastat situace, kdy budete potřebovat pomoci s nějakým příkazem, nastavením programu, či uvedením nějakého kousku hardwaru do chodu. Naštěstí tu je několik možností, jak pomoc získat. Pokud jste si nainstalovali balíčky ze skupny F (viz dále – instalace), máte již velké množství nápovědy naistalováno. Programy samotné se instalují do systému i s nápovědou týkající se jejich použití, parametrů a konfiguračních souborů. Konečně můžete obdržet pomoc i na oficiálních stránkách Slackware.

Systémová nápověda

man

man (zkratka pro "manuál") je tradiční formou online dokumentace v Unixech a v linuxových operačních systémech. Speciálně formátované soubory, "man pages" (manuálové stránky), jsou vytvořeny pro většinu příkazů a jsou distribuovány společně se softwarem. Spuštěním **man nějaký_příkaz** se zobrazí manuálová stránka pro tento příkaz či program.

Poněvadž manuálových stránek je velké množství, byly seskupeny do očíslovaných sekcí. Tento systém je už tak zavedený, že můžete často vidět odkazy na mauálové stránky různých programů včetně uvedení čísla příslušné sekce. Například jste už možná viděli zápis: **man(1)**. To vám říká, že příkaz **man** je dokumentován v sekci 1. Můžete sami specifikovat číslo sekce pro žádaný příkaz; například můžete napsat **man 1 man**. Uvedení čísla sekce, kterou má příkaz man prohledávat je užitečné v případě výskytu vícero různých položek s tím samým jménem v různých sekcích.

Tabulka 2–1. Sekce manuálových stránek

Sekce	Obsah
Sekce 1	uživatelské příkazy (pouze úvod)
Sekce 2	systémová volání
Sekce 3	volání C knihovny
Sekce 4	zařízení (např: hd, sd)
Sekce 5	formáty souborů a protokoly (např: wtmp, /etc/passwd, nfs)
Sekce 6	hry (jen úvod)
Sekce 7	konvence, balíčky maker, atd. (např: nroff, ascii)
Sekce 8	správa systému (jen úvod)

Kromě příkazu **man**(1) ještě existují příkazy **whatis**(1) a **apropos**(1), jejichž společným cílem je učinit snadnějším vyhledávání informací v systému man. **whatis** poskytuje velmi stručný popis systémových příkazů ve stylu kapesní referenční příručky. **apropos** je používán k nalezení manuálové stránky obsahující dané klíčové slovo.

Podrobnosti hledejte v manuálových stránkách.

Adresář /usr/doc

Většina balíčků které sestavujeme, přichází s nějakým druhem dokumentace: Soubory README, instrukce k používání, licenční soubory... Jakýkoliv druh dokumentace, který přichází společně se zdrojem je uložen do vašeho systému, do adresáře `/usr/doc`.

Pokud manuálové stránky neposkytnou dostatečné informace, pak by `/usr/doc` měl být vaším dalším krokem.

HOWTO a mini–HOWTO

Je obrazem pravého ducha komunity, že vám přináší kolekci HOWTO/mini–HOWTO. Tyto soubory jsou přesně tím, co naznačuje název, tedy dokumenty popisujícími jak něco udělat. Pokud si nainstalujete sbírku balíčků HOWTO, pak je najdete v adresáři `/usr/doc/Linux-HOWTOs` a mini–HOWTO pak v: `/usr/doc/Linux-mini-HOWTOs`.

V tom samém balíčku je přiložena i sbírka FAQ (Frequently Asked Questions – často kladené otázky) – otázky včetně odpovědí. Instalují se tamtéž.

Číst tyto soubory má opravdu cenu kdykoliv si nejste zcela jisti jak si s něčím poradit. Udivující rozsah témat je často rozepsán do překvapivých podrobností.

Kapitola 3. Instalace

Obsah

Abyste mohli Slackware Linux používat, musíte jej získat a nainstalovat. Získat Slackware můžete snadno tak, že si jej objednáte, nebo zdarma stáhnete přes Internet. Jeho instalace je rovněž snadná, máte-li základní vědomosti o vašem počítači a máte chuť se ještě něco trochu přiučit. Instalační program samotný vás vede krok za krokem. Z těchto důvodů to můžete mít všechno pohromadě a provozovat velmi rychle.

Získání Slackwaru

Oficiální disky a krabicová balení

Oficiální sadu CD Slackware Linuxu dodává Slackware, Inc. Objednáním oficiální sady disků získáváte výhodu CD instalace, mailovou instalační podporu, 30ti stránkovou instalační knížečku a další. Krabicová souprava obsahuje sadu CD plus oficiální manuál k Slackware Linuxu. Možná nejdůležitější je to, že objednání diskových sad je výborný způsob, jak bezprostředně podpořit projekt Slackware Linuxu.:o)

Tabulka 3-1. Kontaktní informace na Slackware, Inc.

Metoda	Informace
telefon	1-800-786-9907
website	http://www.slackware.com
email	<orders@slackware.com>
snail mail	4041 Pike Lane, Suite F Concord, CA 94520-1207

Přes internet

Slackware Linux je rovněž zdarma dostupný přes Internet. Můžete mailovat své instalační dotazy, ale vyšší prioritu v odpovídání mají ti, kdo si objednali oficiální sadu CD.

Oficiální website projektu Slackware Linux je umístěná na:

http://www.slackware.com/

Hlavní lokace FTP Slackware Linuxu je:

ftp://ftp.slackware.com/pub/slackware/

Kapitola 4. Konfigurace systému

Obsah

Dřív než budete moci nastavovat složitější části systému, bude dobré se naučit něco o tom, jak je systém organizován a jaké příkazy je možné používat k vyhledávání souborů a programů. Je také dobré vědět, jaké kroky je třeba podniknout, když si budete chtít vytvořit jádro podle svých představ. Tato kapitola vás obeznámí s organizací systému a s konfiguračními soubory. Pak už se můžeme dostat k nastavování i složitějších částí systému.

Prohlídka systému

Ještě než se vrhneme na různé možnosti nastavovení, je důležité pochopit jak je linuxový systém poskládán. Linuxový systém je významně odlišný od DOSu či Windows (jakož i Macintoshe), takže v této části knihy vám pomůžeme ponořit se do jeho struktury, abyste si mohli později snáze nastavit váš systém tak, aby se to potkalo s vašimi potřebami.

Struktura souborového systému

První výrazný rozdíl mezi Slackware Linuxem a DOSem či Windows najdeme v uspořádání souborového systému. Pro začátečníky: Nepoužívají se písmena k označování diskových oddílů. Pod Linuxem máme jeden hlavní adresář. Můžete si ho představit třeba jako C: pod DOSem. Každý oddíl ve vašem systému je mountován (připojován) do adresáře v onom hlavním adresáři. Vypadá to jako nějaký stále se nafukující harddisk.

Tomuto hlavnímu adresáři říkáme "root" (kořenový) adresář a označujeme jej prostě lomítkem (`/`). Toto pojetí možná vypadá podivně, ale ve skutečnosti vám velmi usnadňuje život v situaci, kdy chcete diskový prostor rozšířit. Například se vám stane, že už se nevejdete na disk, kde máte připojený adresář `/home`. Většina lidí instaluje Slackware s jediným obrovským root systémem. Takže, když diskový oddíl může být připojen pod libovolný adresář, můžete jednoduše skočit do obchodu, koupit tam další harddisk a namountovat (připojit) si jej pod adresář `/home`. No a máte "naroubováno" o něco více prostoru do vašeho systému. A to vše bez nutnosti hýbat s mnoha věcmi okolo.

Dále vám popíšeme hlavní adresáře nacházející se na nejvrchnější úrovni souborového systému Slackwaru.

/bin
: Zde jsou uloženy ty nejzákladnější uživatelské programy. Ty představují nejminimálnější sadu programů nutných k tomu, aby uživatel mohl systém používat. Jsou tu uloženy takové věci, jako je shell, příkazy souborového systému (`ls`, `cp`) a podobně. Adresář `/bin` se po nainstalování systému už zpravidla nemění. Pokud ano, tak obvykle jen z důvodu upgradu softwarových balíčků.

/boot
: Zde jsou soubory, které používá Linux Loader (LILO). I v tomto adresáři se toho po nainstalování systému už moc nemění.

/cdrom
: Pamatujete si ještě, že všechna zařízení se musí připojovat do nějakého adresáře pod hlavním root adresářem? Dobrá, takže `/cdrom` se používá jako přípojný bod (mount point) pro vaši CD–ROM jednotku.

/dev
: Se všemi věcmi v Linuxu se zachází jako se soubory. Takže i s hadrwarovými zařízeními jako např. sériovými porty, harddisky a scanery. Abychom mohli k těmto zařízením přistupovat, musí být přítomen speciální soubor nazývaný "device node". Všechny "device nodes" jsou uloženy v adresáři `/dev`. Zjistíte, že takhle je to uděláno ve většině operačních systémů Unixového typu.

/etc
: V tomto adresáři jsou uloženy systémové konfigurační soubory. Všechno od konfigruačních souborů pro X Window, přes uživatelské databáze až po systémové startovací skripty je zde. Systémový administrátor se časem s tímto adresářem hodně sžije.

/home
: Linux je víceuživatelský operační systém. Každému uživateli je v systému zřízen účet a jedinečný adresář pro jeho osobní soubory. Tento adresář je nazýván jako uživatelův domovský (home). Adresář `/home` je používán jako defaultní lokace pro umístění uživatelských domovských adresářů.

/lib
: Zde jsou uloženy systémové knihovny, které jsou vyžadovány pro základní operace. Najdeme tu například knihovnu jazyka C, dynamický zavaděč, knihovnu ncurses, moduly jádra, atd.

/lost+found
: Při startu systému probíhá i kontrola souborových systémů na chyby. Jsou–li chyby zjištěny, spustí se program **fsck**, aby zjistil, zda půjdou opravit. Opravené části souborového systému jsou zapsány do adresáře `/lost+found`.

/mnt
: Tento adresář je používán jako dočasný přípojný bod (mount point) pro práci na pevných discích či vyjímatelných médiích.

/opt
: Volitelné (optional) softwarové balíčky. Hlavní myšlenkou pro `/opt` je, aby se každý softwarový balíček instaloval do `/opt/<software package>`, což usnadňuje jeho pozdější odstranění. Slackware umísťuje nějaké věci do `/opt` (např. KDE dává do `/opt/kde`). I vy si můžete do `/opt` svobodně přidávat co chcete.

/proc
: Toto je velmi zvláštní adresář. Ve skutečnosti není částí souborového systému, ale je virtuálním souborovým systémem, který poskytuje přístup k informacím jádra. Různé informace, které vám chce kernel dávat na vědomí, jsou vám dávány prostřednictvím "souborů" v adresáři `/proc`. Rovněž vy můžete zasílat informace do kernelu prostřednictvím některých těchto "souborů". Vyzkoušejte si třeba **`cat /proc/cpuinfo`**.

/root
: Správce systému – administrátor – je systému znám pod jménem "root". Rootův domovský adresář je v `/root` místo v `/home/root`. Důvod k tomu je jednoduchý. Co kdyby `/home` byl na jiném diskovém oddílu než `/` a nemohl by být připojen? Root by se měl přirozeně přihlásit do systému a opravit tento problém. Kdyby byl jeho domovský adresář na poškozeném filesystému, asi by mu to přihlašování hodně zkomplikovalo.

/sbin
: Zde jsou programy, které může spouštět jen root a nebo které jsou spouštěny v průběhu startu systému. Normální uživatelé nemohou spouštět programy z tohoto adresáře.

/tmp
: Místo pro umisťování dočasně existujících souborů. Všichni uživatelé mohou z tohoto adresáře číst i zapisovat do něj.

/usr

Toto je největš í adresář v linuxovém systému. Vš echno ostatní musí jít hezky sem: Programy, dokumentace, zdrojový kód kernelu a X Window systém. Větš inu programů budete instalovat do tohoto adresáře.

/var
Systémové logovací soubory, cache data a programové zámky jsou ukládány sem. Toto je adresář pro často se měnící data.

Nyní byste měli mít dobrý přehled o tom, co který adresář v souborovém systému obsahuje. V následující sekci vám řekneme, jak lze jednoduš e nalézt nějaký soubor tak, abyste to nemuseli dělat ručně.

Vyhledávání souborů

Teď už víte, co který adresář obsahuje, ale při hledání souborů vám to až tak moc nepomůž e. Jistě, můž ete se probírat jednotlivými adresáři, ale jsou tu i rychlejš í cesty. V Slackwaru máme čtyři základní příkazy pro vyhledávání souborů.

which

Prvním z nich je příkaz **which**(1). Obvykle je použ íván k rychlému zjiš tění umístění nějakého programu. Prohledává jen adresáře uvedené "v cestě" `PATH` a jako výsledek vrací první výskyt hledaného souboru a adresářovou cestu k němu. Příklad:

```
$ which bash
/bin/bash
```

Tady vidíte, ž e `bash` je v adresáři `/bin`. Toto je velmi omezený příkaz, protož e prohledává jen adresáře v `PATH`.

whereis

Příkaz **whereis**(1) pracuje podobně jako **which**, ale navíc můž e vyhledávat man stránky a zdrojové soubory. **whereis** vyhledávání soborů bash by mělo dát tento výsledek:

```
$ whereis bash
bash: /bin/bash /usr/bin/bash /usr/man/man1/bash.1.gz
```

Tento příkaz nám neříká jen, kde je hledaný program, ale též kde je ulož ena online dokumetace k němu. Rovněž tento příkaz je omezený. Co kdybyste chtěli nalézt nějaký konfigurační soubor? Na to nemůž ete použ ít **which** ani **whereis**.

find

Příkaz **find**(1) umí hledat cokoliv. Když bychom například chtěli prohledat celý systém, abychom zjistili, kde se nachází defaultní `xinitrc` soubor, napíš eme:

```
$ find / -name xinitrc
./var/X11R6/lib/xinit/xinitrc
```

Práce příkazu **find** trvá trochu déle, protož e musí prohledat celý adresářový strom. Navíc, pokud jej spouš títe jako normální už ivatel, obdrž íte nejspíš spoustu chybových hláš ení o odmítnutém přístupu do adresářů, které smí prohlíž et jenom root. Ale **find** nám soubor naš el a to je príma. Jenom, kdyby to mohlo být trochu rychlejš í...

locate

Příkaz **locate**(1) prohledává celý souborový systém stejně, jako to dělá příkaz find, ale namísto skutečného souborového systému prohledává svoji databázi. Databáze je nastavena tak, aby se automaticky updatovala ve 4:40 ráno, abyste měli v systému hodně čerstvý seznam souborů. Databázi pro locate můž ete updatovat i ručně spuš těním příkazu **updatedb**(1). Příkaz **updatedb**(1) nemůž e spouš tět normální už ivatel. Tady je příklad příkazu **locate** v akci:

```
$ locate xinitrc
# we don't have to go to the root
# nemusíme být přihlášeni jako root
/var/X11R6/lib/xinit/xinitrc
/var/X11R6/lib/xinit/xinitrc.fvwm2
/var/X11R6/lib/xinit/xinitrc.openwin
/var/X11R6/lib/xinit/xinitrc.twm
```

Dostali jsme víc, než jsme očekávali, a ješ tě k tomu rychle. S těmito příkazy byste měli být schopni nalézt cokoli co chcete.

Adresář /etc/rc.d

Systémové inicializační soubory jsou ulož eny v adresáři `/etc/rc.d`. Slackware pro svoje inicializační soubory použ ívá rozmístění ve stylu BSD. Kaž dý task (úloha) či runlevel má svůj odpovídající rc soubor. To poskytuje organizovanou strukuru, kterou je snadné udrž ovat.

Je několik kategorií inicializačních souborů. Jsou to: system startup, runlevels, inicializace sítě a System V kompatibilita. A – v souladu s tradicí – vš echno ostatní házíme do kategorie "other" (ostatní).

System Startup

Prvním programem, který se spouš tí pod Slackwarem (nepočítáme–li kernel) je **init**(8). Tento program čte soubor `/etc/inittab`(5), aby věděl, jak zprovoznit systém. Dále spouš tí skript `/etc/rc.d/rc.S`, který připraví systém před tím, než přejde do pož adovaného runlevelu. Soubor `rc.S` zpřístupňuje virtuální paměť , připojuje filesystémy, čistí určité logovací adresáře, inicializuje zařízení Plug and Play, zavádí moduly jádra, konfiguruje zařízení PCMCIA, nastavuje sériové porty a spouš tí inicializační skripty pro System V (pokud nějaké najde). Dále `rc.S` volá některé dalš í skripty:

rc.S
Toto je vlastní systémový inicializační skript.

rc.modules
Zavádí moduly jádra. Věci jako je vaš e síť ová karta, podpora PPP, a dalš í věci se zavádějí tady. Pokud tento skript najde soubor `rc.netdevice`, rovněž jej spustí.

`rc.pcmcia`
: Hledá a nastavuje veš kará PCMCIA zařízení, která můž ete ve svém systému mít. To je nejuž itečnějš í pro majitele laptopů, kteří pravděpodobně mají PCMCIA modem nebo síť ový adaptér.

`rc.serial`
: Konfiguruje sériové porty spuš těním přísluš ných příkazů **setserial**.

`rc.sysvinit`
: Hledá System V init skripty pro pož adovaný runlevel a spouš tí je. Detailněji o tom pohovoříme dále.

Runlevel Initialization Scripts

Po dokončení inicializace systému se **init** pustí do inicializace runlevelu. Runlevel popisuje v jakém stavu váš stroj poběž í. Jinak řečeno: Runlevel říká initu jestli budete akceptovat víceuž ivatelské přihlaš ování nebo jen jednouž ivatelský mód, zda chcete nebo nechcete síť ové služ by a jestli se o obsluhu přihlaš ování postará X Window systém, nebo **agetty**(8). Následující soubory definují různé runlevely v Slackware Linuxu.

`rc.0`
: Zastavení systému (runlevel 0). Defaultně je symbolickým linkem spojen na rc.6.

`rc.4`
: Víceuž ivatelský rež im (runlevel 4) v X11 s KDM, GDM, nebo XDM coby přihlaš ovacím manaž erem.

`rc.6`
: Reboot systému (runlevel 6).

`rc.K`
: Start v jednouž ivatelském módu (runlevel 1).

`rc.M`
: Víceuž ivatelský mód (runlevel 2 a 3) se standardním přihlaš ováním v textové konzoli. Toto je standardní runlevel ve Slackwaru.

Inicializace sítě

Runlevely 2, 3 a 4 spouš tějí síť ové služ by. Následující soubory jsou zodpovědné za inicializací sítě:

`rc.inet1`
: Je vytvořen programem **netconfig**. Tento soubor zodpovídá za konfiguraci stávajícího síť ového rozhraní.

`rc.inet2`
: Běž í po `rc.inet1` a spouš tí základní síť ové služ by.

`rc.atalk`
: Spouš tí služ by AppleTalk.

`rc.httpd`
: Spouš tí Apache web server.

`rc.samba`
: Spouš tí služ by pro sdílení souborů a tiskáren s Windows.

`rc.news`
: Spouš tí news server.

System V Compatibility

System V init compatibility byla zavedena v Slackware 7.0. Mnoho jiných linuxových distribucí použ ívá tento styl místo BSD stylu. V podstatě je v System V kaž dému runlevelu přidělen podadresář, kde má umístěny svoje init skripty, zatímco BSD styl dává kaž dému runlevelu po jednom skriptu.

Skript `rc.sysvinit` vyhledává inicializační skripty pro System V, které máte v `/etc/rc.d` a spouš tí je, odpovídá−li to danému runlevelu. To je už itečné pro některé komerční softwarové balíčky, které instalují inicializační skripty jak pro System V, tak i pro BSD styl.

Ostatní soubory

Skripty popisované níž e jsou dalš ími systémovými inicializačními skripty. Jsou typicky spuš těny z jednoho z výš e uvedených hlavních inicializačních skriptů, takž e jediné co potřebujete udělat, je editovat jejich obsah.

`rc.cdrom`
: Je−li to povoleno, tento skript bude hledat CD−ROM v mechanice a namountuje ho pod `/cdrom`, když nějaké najde.

`rc.gpm`
: Spouš tí "general purpose mouse services". Ty umož ňují funkce kopíruj/vlož pomocí myš i v linuxové konzoli.

`rc.ibcs2`
: Spouš tí "Intel Binary Compatibility support". To je potřeba jedině když plánujete spouš tění programů kompilovaných pro SCO UNIX, nebo jiný komerční Intel UNIX. Nepotřebujete to pro spouš tění linuxových programů.

`rc.font`
: Zvádí "custom" obrazovkový font pro konzolu.

`rc.local`
: Obsahuje startovací příkazy specifické pro váš systém. Po instalaci je prázdný a je rezervován pro místní administrátory. Tento skript je spouš těn až po vš ech ostatních inicializacích.

K povolení (enable) skriptu musíte udělat jedinou věc: Nastavit mu práva ke spouš tění pomocí příkazu **chmod**. Zakázání spouš tění skriptu dosáhnete odebráním spouš těcích práv. Více informací o příkazu **chmod** najdete v sekci *Permissions* v 9. kapitole.

Kapitola 5. Konfigurace sítě

Obsah

Síťový Hardware

Jako pro většinu zajímavých věcí, které můžete s počítačem dělat, i pro jeho připojování k síti budete potřebovat nějaký speciální hardware. Budete potřebovat nějakou NIC (Network Interface Card – kartu síťového rozhraní) pro připojení do LAN, a možná taky modem pro připojování k nějakému internetovému providerovi, nebo možná oboje (nebo několik toho, nebo nic).

Podle způsobu konfigurace můžeme rozdělit hardware na kategorie PCMCIA (pro laptopy) a ne–PCMCIA. Důvodem pro toto možná poněkud podivné dělení je, že v současnosti není PCMCIA hardware podporován přímo jádrem (distribucí jádra), ale odděleným balíčkem, který obsahuje nezbytné ovladače (jako moduly jádra) a nějaký software pro nastavení a správu zařízení PCMCIA. Všechno ostatní samozřejmě zvládá standardní distribuce jádra.

netmods

Ovladače síťových zařízení, která kernel podporuje, jsou uložena v balíčku netmods (`slakware/n3/netmods.tgz`). Pokud ještě nemáte netmod nainstalován, budete to muset udělat teď. (Viz Kapitola 16 pro pomoc s instalováním softwarových balíčků.)

Moduly jádra, které se mají zavádět při bootování jsou zváděny ze souboru `rc.modules`, který najdeme v `/etc/rc.d.`. Defaultní soubor `rc.modules` obsahuje sekci "Network device support". Otevřete–li `rc.modules` a podíváte se do této sekce, všimnete si, že nejdříve testuje spustitelný soubor `rc.netdevice` umístěný v `/etc/rc.d`. Soubor `rc.netdevice` je vytvořen během instalace, jestliže **setup** úspěšně najde (automatickým zkoušením) nějaká síťová zařízení. Pokud tedy ano, nejspíš tohle číst nemusíte (ach, paradox); pokud ale ne, pak čtěte dál.

Pod tímto "if" blokem je seznam síťových zařízení a řádků modprobe. Všechny zakomentářované. Najděte svoje zařízení a odkomentářujte odpovídající modprobe řádek. Potom soubor uložte. Teď – jako root – spusťte `rc.modules`. To by mělo zavést ovladače síťových zařízení (a rovněž jakékoliv další moduly, které jsou v seznamu a jsou odkomentářované). Povšimněte si, že některé moduly (jako třeba ne2000) vyžadují parametry; ujistěte se, že vybíráte správný řádek.

PCMCIA síťová zařízení

PCMCIA síťová zařízení by měla být ještě snadnější než ostatní. Ujistěte se, že máte balíček pcmcia (`slakware/a11/pcmcia.tgz`) nainstalovaný. (viz Chapter 16 pro podrobnosti o instalování balíčků.) Při instalaci vytvoří balíček pcmcia soubor `rc.pcmcia` v adresáři `/etc/rc.d`, dále adresář `/etc/pcmcia` a nainstaluje ovladače do `/lib/modules/<kernel version>/pcmcia`. Velice "cool" věcí kolem pcmcia balíčku je, že se bude pokoušet autodetekovat vkládání a vyjímání podporovaných pcmcia zařízení. Jednoduše zkuste vložit váš pcmcia síťový adaptér a poslouchejte jak pípne, když zavádí nezbytné moduly. Když kartu vyjmete, její ovládací moduly by měly být automaticky vyjmuty rovněž.

Bohužel, když zkompilujete novější verzi jádra, budete muset pravděpodobně rekompilovat pcmcia–cs, aby byly ovladače updatovány. Samozřejmě, že zdrojový kód je přiložen; koukněte do adresáře `source/a/pcmcia`. Tam zdroj, skripty a nějakou dokumenteci najdete.

Kapitola 6. X Window Systém

Obsah

X Window System je standardním GUI (grafické uživatelské rozhraní) na všech UNIXových platformách, tedy i na Linuxu. Narozdíl od Windows či MacOS je GUI v Linuxu odděleno od hlavního operačního systému jádra. To systému přidává na stabilitě: Jestliže GUI zhavaruje, nevezme s sebou i celý zbytek systému.

Jedním problémem s X je ten, že už tradičně je poměrně obtížně konfigurovatelný. Proto Slackware 7 uvedl nekonfigurovací nastavení pro X, které používá ovladač framebufferu. To znamená, že nemusíte procházet procedurami popsanými v sekcích *xf86config* a *XF86Setup*. Framebuffer bude pracovat na veškerých video kartách, které splňují standard VESA 2.0. To značí, že všechny moderní video karty budou pod X fungovat. Na druhou stranu je framebuffer znatelně pomalejší než konfigurace X ušitá vašemu systému na míru.

Pokud se rozhodnete používat framebuffer server, budete muset nainstalovat balíček `xxfb.tgz` ze softwarové skupiny X. Rovněž byste si měli během instalace vybrat rozlišení pro konzoli. Doporučená volba pro X je pro většinu lidí obvykle tou nejlepší.

Rozhodnete-li se konfigurovat X pro váš systém, budet potřebovat projít následující instrukce v sekcích *xf86config*, nebo *XF86Setup*. První sekce popisuje používání programu **xf86config**(1), což je program pro konfiguraci X, pracující v příkazové řádkce. Druhá sekce popisuje program **XF86Setup**(1), což je grafická verze konfiguračního programu.

xf86config

xf86config je jedním ze dvou programů, které mohou být použity pro nastavování X na vašem systému. Základní idea je jednoduchá: Budete počastováni řadou dotazů a možností jak odpovědět. Vybírejte odpovědi, které vašemu systému padnou nejlépe. Po projití celým programem bude vytvořen soubor `/etc/XF86Config`(5) a váš systém budete připraven na spuštění X. Pokud se během dotazování někde spletete, budete muset program ukončit použitím **`control-c`** a začít znovu od začátku.

Velice pomůže, když si zjistíte maximum informací o vašem monitoru a video kartě ještě dřív, než **xf86config** spustíte. Získat informace o videokartě vám pomůže rovněž program **SuperProbe**:

```
# SuperProbe
```

Ten vám nejdříve vypíše varování o možném uzamčení (zatuhnutí) systému. Pokud vás to vyděsí, máte pět vteřin na to, abyste stiskem **`control-c`** testování zabránili. V opačném případě se dočkáte informací o nastavení vaší video karty:

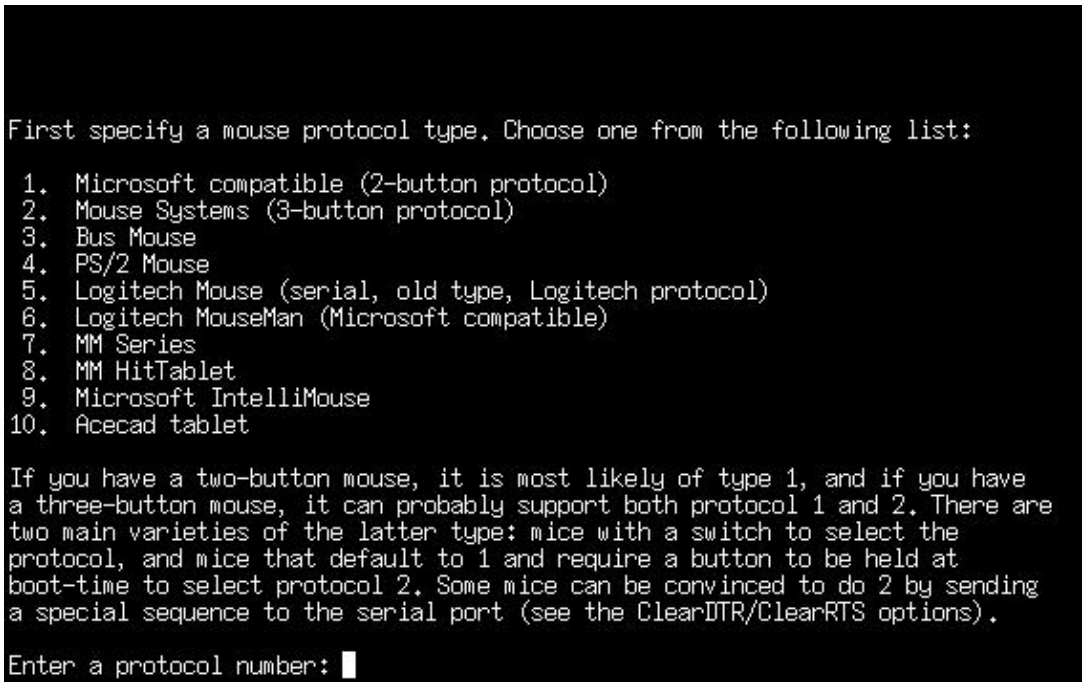
```
        First video: Super-VGA
        Chipset: ATI 264GT3 (3D Rage Pro) (Port Probed)
        Memory: 4096 Kbytes
        RAMDAC: ATI Mach64 integrated 15/16/24/32-bit
                DAC w/ clock
                (with 8-bit wide lookup tables)
                (programmable for 6/8-bit wide lookup tables)
        Attached graphics coprocessor:
                Chipset: ATI Mach64
                Memory: 4096 Kbytes
```

Tak takhle nějak vypadají informace o kartě ATI Rage Pro. Zapište si tyto informace, a nebo se přepněte na další virtuální terminál (použitím klávesové zkratky **`alt-Fx`**) a spuťte **xf86config**. Informace o videokartě budete později potřebovat. Program **xf86config** musíte spouštět jako root, protože bude zapisovat soubory a vytvářet symbolické linky v místech, kde je dovoleno zapisovat pouze rootovi:

```
# xf86config
```

Po spuštění vás **xf86config** uvítá obrazovkou plnou textu popisujícího, co všechno se chystá provádět. Pamatujte, neexistuje možnost vracet se zpět na předchozí obrazovku uděláte-li chybu, takže odpovědi vybírejte pečlivě. Jinak to budete muset dělat několikrát. Takže teď stiskněte **`enter`**, jak vás o to žádá.

Protokol myši

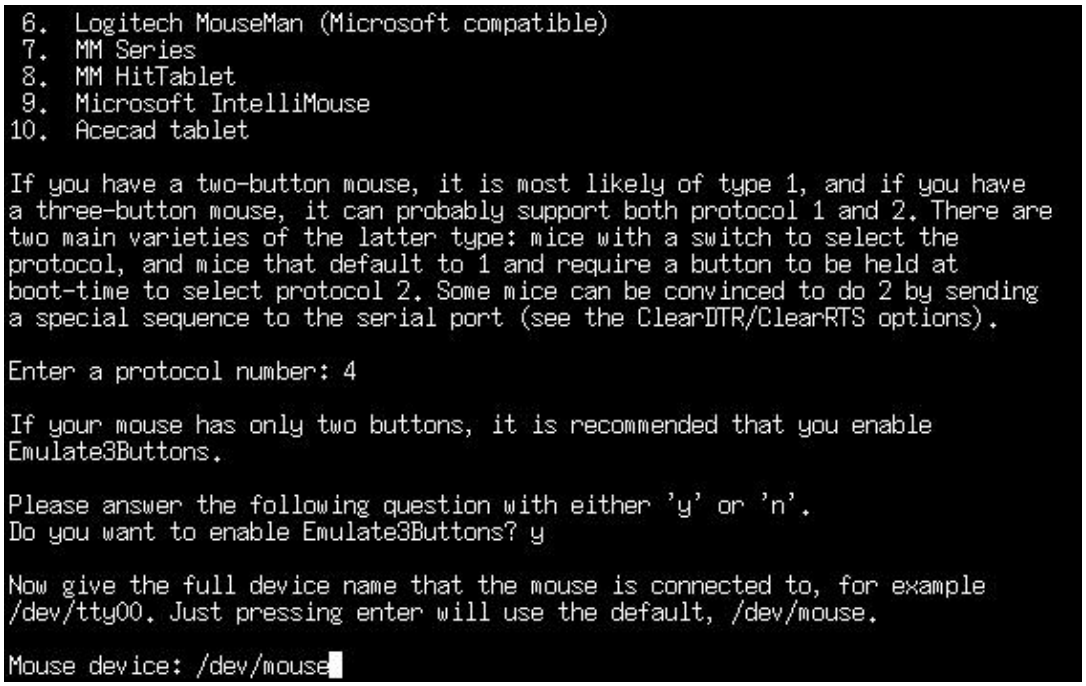


Vyberte si ze seznamu typ myš i, který máte. V současnosti větš ina myš ek bude PS/2 nebo Microsoft Intellimouse. Starš í myš i budou pravděpodobně vyž adovat nějaký jiný uvedený typ.

Emulace 3. tlačítka

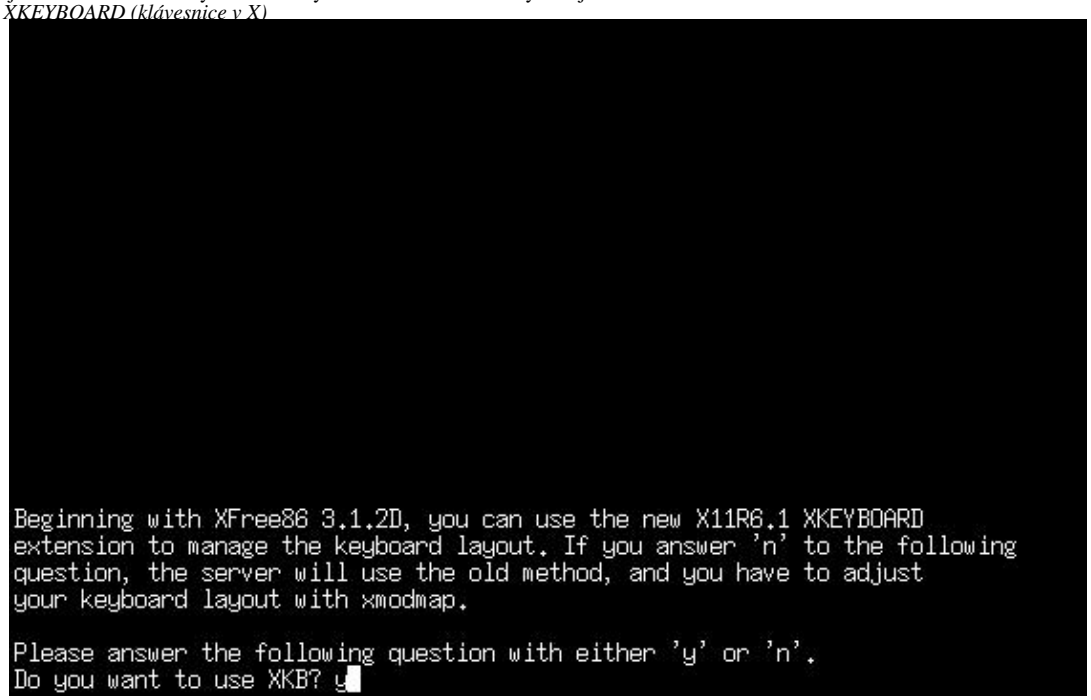
Pokud máte na myš i jen dvě tlačítka, můž ete si vyž ádat emulaci třetího tlačítka. Kliknutí obou myš ích tlačítek současně bude interpretováno jako stisk třetího tlačítka. Protož e mnoho programů využ ívá třetího tlačítka, je tato emulace doporučována. Máte–li třítlačítkovou myš , emulaci nepotřebujete.

Jméno zařízení myš i



Obvykle vyhoví defaultní `/dev/mouse`. Ovš em máte–li myš zapojenou do nějakého zvláš tního portu, mož ná budete muset zadat něco jiného. Větš ině sériových a PS/2 myš í defaultní nastavení vyhovuje.

Rozš íření XKEYBOARD (klávesnice v X)



Pravděpodobně budete chtít použ ívat rozš íření X klávesnice. Pokud to nevyberete, dočkáte se podivného chování kláves backspace a delete. Vybráním rozš íření zajistíte, ž e se tyto klávesy budou chovat tak jak mají.

Vazby na klávesy alt (Bindings for alt keys)

Chcete–li zadávat znaky z jiných jazyků než angličtiny, měli byste vázání kláves alt povolit. Chystáte–li se psát pouze angličtinu, tyto vazby potřebovat nebudete.

Rozsah horizontální synchronizace (Horizontal sync range)

```
You must indicate the horizontal sync range of your monitor. You can either
select one of the predefined ranges below that correspond to industry-
standard monitor types, or give a specific range.

It is VERY IMPORTANT that you do not specify a monitor type with a horizontal
sync range that is beyond the capabilities of your monitor. If in doubt,
choose a conservative setting.

   hsync in kHz; monitor type with characteristic modes
 1  31.5; Standard VGA, 640x480 @ 60 Hz
 2  31.5 - 35.1; Super VGA, 800x600 @ 56 Hz
 3  31.5, 35.5; 8514 Compatible, 1024x768 @ 87 Hz interlaced (no 800x600)
 4  31.5, 35.15, 35.5; Super VGA, 1024x768 @ 87 Hz interlaced, 800x600 @ 56 Hz
 5  31.5 - 37.9; Extended Super VGA, 800x600 @ 60 Hz, 640x480 @ 72 Hz
 6  31.5 - 48.5; Non-Interlaced SVGA, 1024x768 @ 60 Hz, 800x600 @ 72 Hz
 7  31.5 - 57.0; High Frequency SVGA, 1024x768 @ 70 Hz
 8  31.5 - 64.3; Monitor that can do 1280x1024 @ 60 Hz
 9  31.5 - 82.0; Monitor that can do 1280x1024 @ 76 Hz
10  31.5 - 95.0; Monitor that can do 1280x1024 @ 85 Hz
11  Enter your own horizontal sync range

Enter your choice (1-11): 4
```

Tohle je první z řady otázek týkajících se monitoru. Je důležité, abyste tady vybírali opravdu moudře. Nezadávejte rozsah, který je mimo specifikace vašeho monitoru. U nových monitorů to tak důležité není, protože ty se nebudou pokoušet dělat nic, co by bylo mimo rozsah jejich specifikací. Ověřením starších monitory se mohou poškodit. Jste-li na pochybách, zvolte konzervativnější rozsah.

Dokumentace k vašemu monitoru bude dobrým referenčním zdrojem pro několik následujících otázek. Pro většinu novějších monitorů můžete nejspíš vybrat 31.5–48.5, nebo 31.5–57.0. Máte-li high-end monitor, můžete vybrat některý z vyšších rozsahů. Nebo si můžete zadat vlastní rozsah horizontální synchronizace, nevidíte-li žádný, který by se vám hodil.

Rozsah vertikální synchronizace (Vertical sync range)

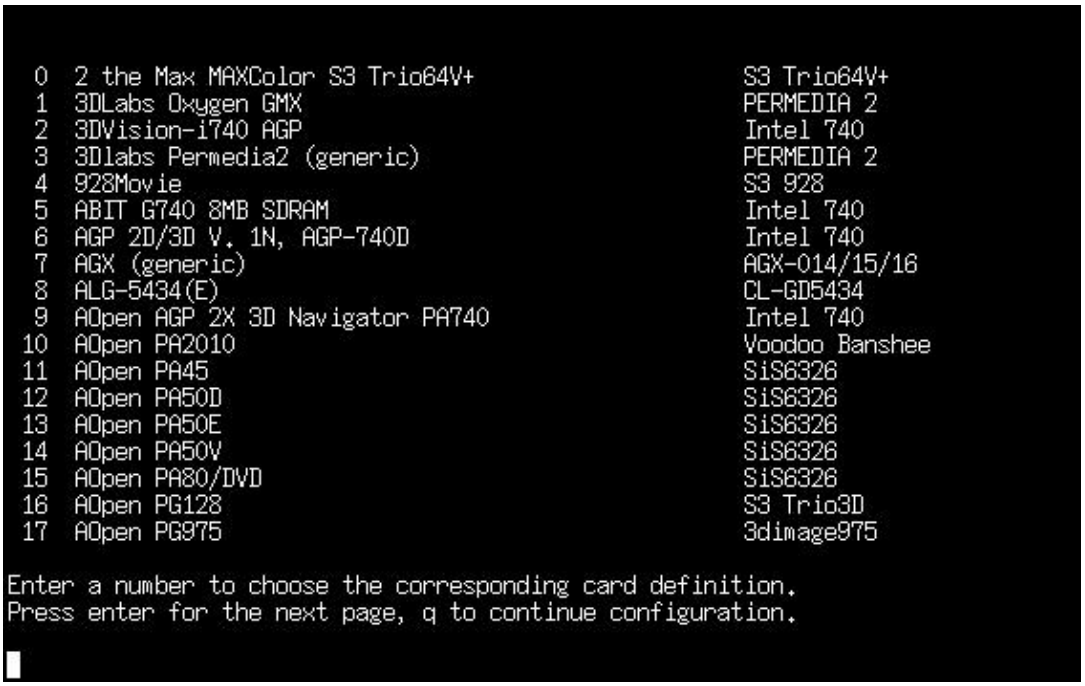
```
 2  31.5 - 35.1; Super VGA, 800x600 @ 56 Hz
 3  31.5, 35.5; 8514 Compatible, 1024x768 @ 87 Hz interlaced (no 800x600)
 4  31.5, 35.15, 35.5; Super VGA, 1024x768 @ 87 Hz interlaced, 800x600 @ 56 Hz
 5  31.5 - 37.9; Extended Super VGA, 800x600 @ 60 Hz, 640x480 @ 72 Hz
 6  31.5 - 48.5; Non-Interlaced SVGA, 1024x768 @ 60 Hz, 800x600 @ 72 Hz
 7  31.5 - 57.0; High Frequency SVGA, 1024x768 @ 70 Hz
 8  31.5 - 64.3; Monitor that can do 1280x1024 @ 60 Hz
 9  31.5 - 82.0; Monitor that can do 1280x1024 @ 76 Hz
10  31.5 - 95.0; Monitor that can do 1280x1024 @ 85 Hz
11  Enter your own horizontal sync range

Enter your choice (1-11): 4

You must indicate the vertical sync range of your monitor. You can either
select one of the predefined ranges below that correspond to industry-
standard monitor types, or give a specific range. For interlaced modes,
the number that counts is the high one (e.g. 87 Hz rather than 43 Hz).

 1  50-70
 2  50-90
 3  50-100
 4  40-150
 5  Enter your own vertical sync range

Enter your choice: 2
```

Ještě jednou: Budete potřebovat znát specifikace vašeho monitoru, abyste mohli odpovědět na tuto otázku. Jste-li na pochybách, vyberte menší rozsah. Bezpečnou volbou by měla být 50–90 nebo 50–100. Pokud nevidíte rozsah, který by vyhovoval vašemu monitoru, můžete jej zadat ručně.

Identifikační řetězce

Nyní budete ve třech dotazech požádáni o zadání identifikačních řetězců pro váš monitor. Nejsou nijak strašně důležité. Můžete to jenom odentrovat. A nebo si to můžete pojmenovat jakkoliv chcete. Tyto řetězce budou použity v konfiguračním souboru pro identifikační účely.

Databáze video karet



Následující část konfigurace X se zabývá vaš í videokartou. Teď se vám bude velmi hodit dokumentace ke kartě a rovněž informace získané pomocí **SuperProbe**. Na úvodní otázku odpovězte "y", protož e budete potřebovat probrat se databází videokaret, abyste tam naš li tu vaš i. Pouhé odentrování by prohledávání databáze přeskočilo a vy byste se dostali rovnou do dalš í části konfigurační procedury.

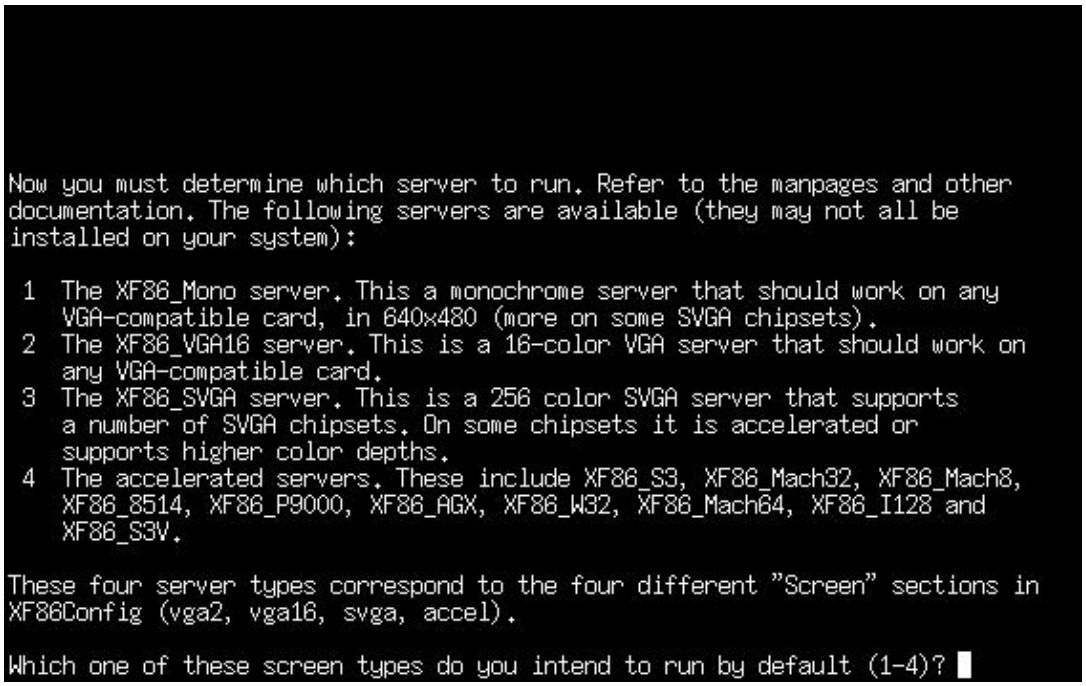
V databázi je přes 800 karet. V levém sloupci je pořadové číslo karty a její název. V pravém sloupci je uveden chipset této karty. Mačkejte enter tak dlouho, dokud v seznamnu nenajdete vaš i videokartu. Až ji najdete, napiš te její číslo a stiskněte enter. Pokud nevíte jaký druh videokarty máte, je tu několik možností co s tím: Zaprvé se můž ete podívat na výpis programu **SuperProbe**, jaký "chipset" tam je uveden a pokusit se najít v databázi kartu s odpovídajícím chipsetem. Nebo můž ete použ ít typ generic SVGA. Mnoho karet, které nemají svůj vlastní server, je podporováno SVGA serverem, takž e toto by měla být bezpečná volba.

Po vybrání karty se vám dostane několika doplňujících informací. Například pro výš e zmiňovanou ATI Rage Pro jsou to tyto informace:

```
Your selected card definition:

Identifier: ATI Mach64
Chipset:    ATI-Mach64
Server:     XF86_Mach64
Do NOT probe clocks or use any Clocks line.
```

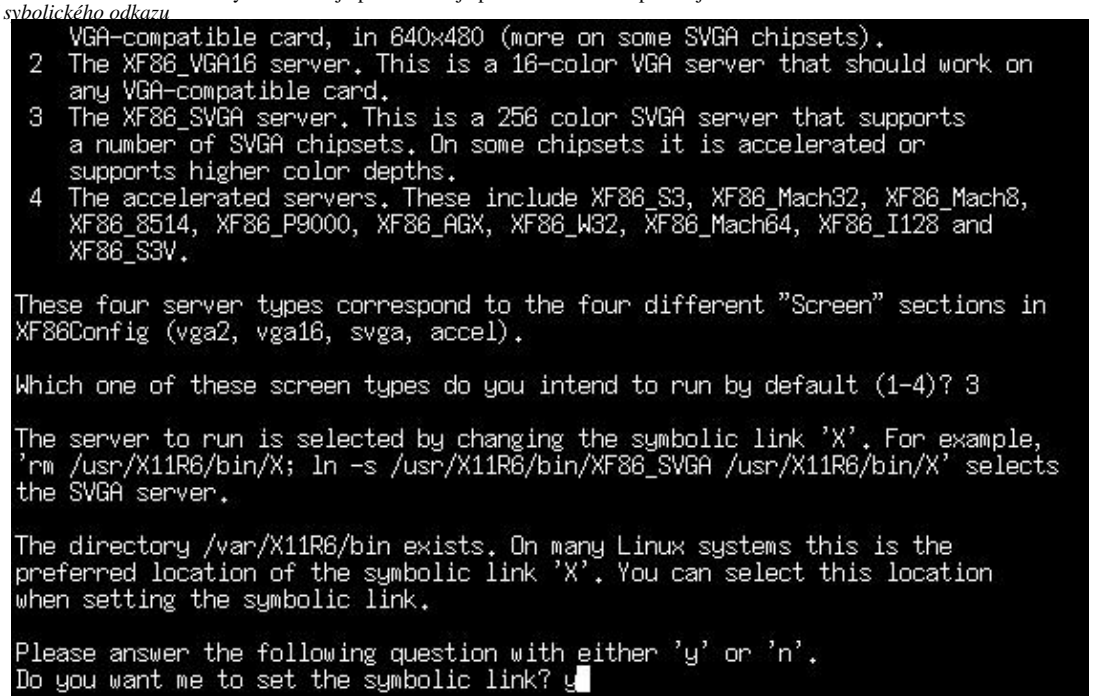
V tomto bodě byste měli pro jistotu zkontrolovat, zda máte nainstalovaný serverový balíček. XF86_Mach64 server je v balíčku `xma64.tgz`. Ověřte si, zda je nainstalovaný správný serverový balíček, protož e jinak nebudou X schopna běž et.

Který server provozovat?



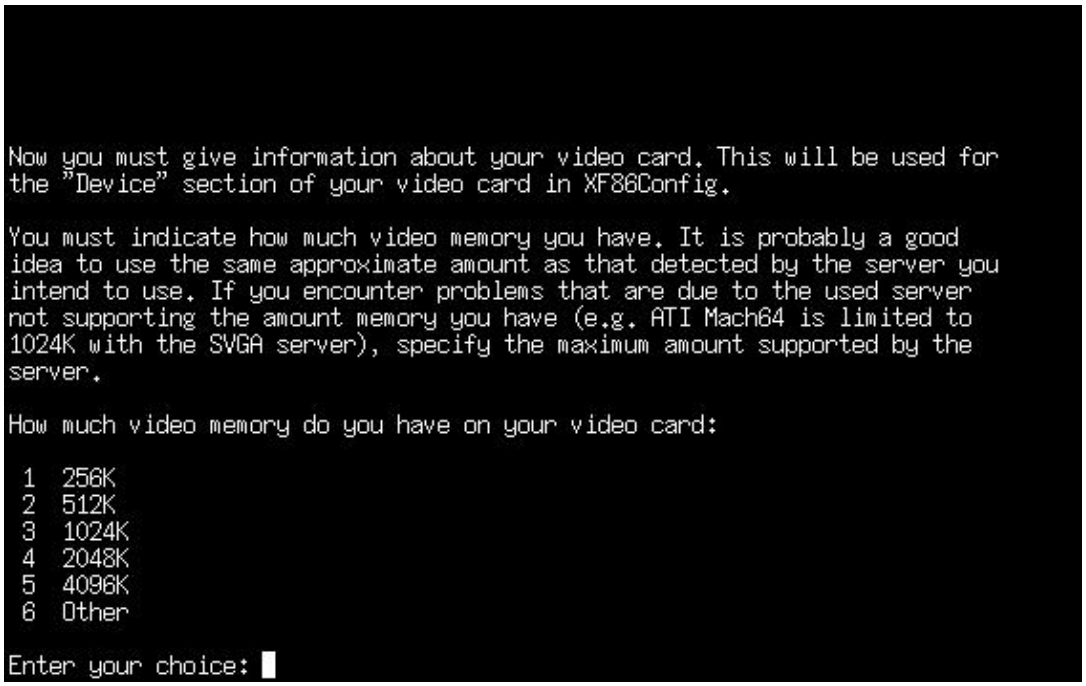
Tato otázka nabízí několik serverů, které můžete používat. Pokud jste správně vybrali videokartu, můžete bezpečně stisknout enter. Tím řeknete X–kám, aby používaly ten server, který odpovídá kartě. Jinak si můžete vybrat k používání Mono server, VGA16 server, SVGA server a nebo akcelerovaný server. Nejlepší volbou je používat server odpovídající kartě.

Vytvoření sybolického odkazu



Pro vytvoření sybolického odkazu odpovězte "y". To vytvoří odkaz k přísluš nému X serveru.

Video paměť



Vyberte, jakou velikost paměti vaš e karta má. K získání této informace můž e poslouž it **SuperProbe**. Máte−li velikost jinou, než nabízejí vypsané volby, můž ete vybrat "Other" a zadat jinou velikost. Dejte pozor, abyste velikost paměti uvedli v kilobajtech.

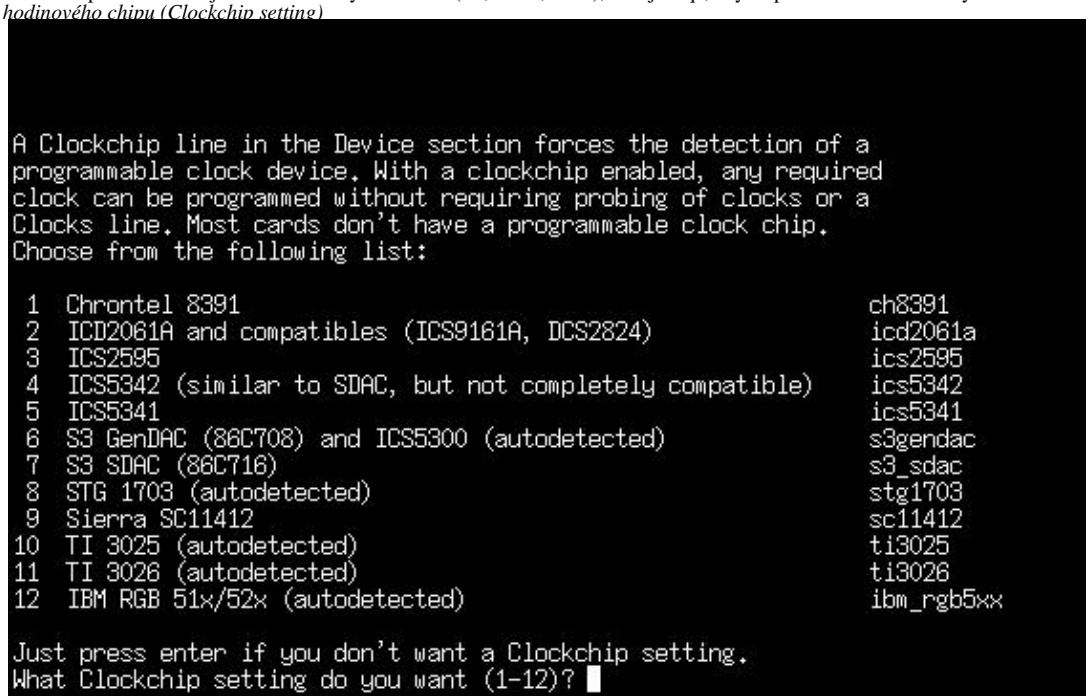
Identifikační řetězce

Budete vyzváni, abyste zadali dalš í tři identifikační řetězce. Tyto se vztahují k vaš í videokartě. Jako u monitoru i zde je zcela v pořádku pouze odentrovat vš echny tři otázky, jestliž e nechcete svou videokartu nijak pojmenovávat.

RAMDAC

Vybírat nastavení RAMDAC budete potřebovat jedině v případě, pokud použ íváte servery S3, AGX nebo W32. **SuperProbe** vám řekne, jaký druh RAMDAC chipu je na vaš í videokartě. Projděte seznamem dokud nenajdete správný chip, potom zadejte odpovídající číslo. Pokud nepoužíváte ani jeden ze zmíněných serverů (S3, AGX, W32), zadejte "q", aby se pokračovalo dál bez vybírání RAMDAC.

Nastavení hodinového chipu (Clockchip setting)



Má−li vaš e karta programovatelný hodinový chip, budete potřebovat vybrat jeden z následujícího seznamu. Větš ina karet nemá programovatelný hodinový chip, takž e by mělo stačit odpovědět jen stiskem entru. **SuperProbe** by vám řekl, jestli má vaš e karta tento chip.

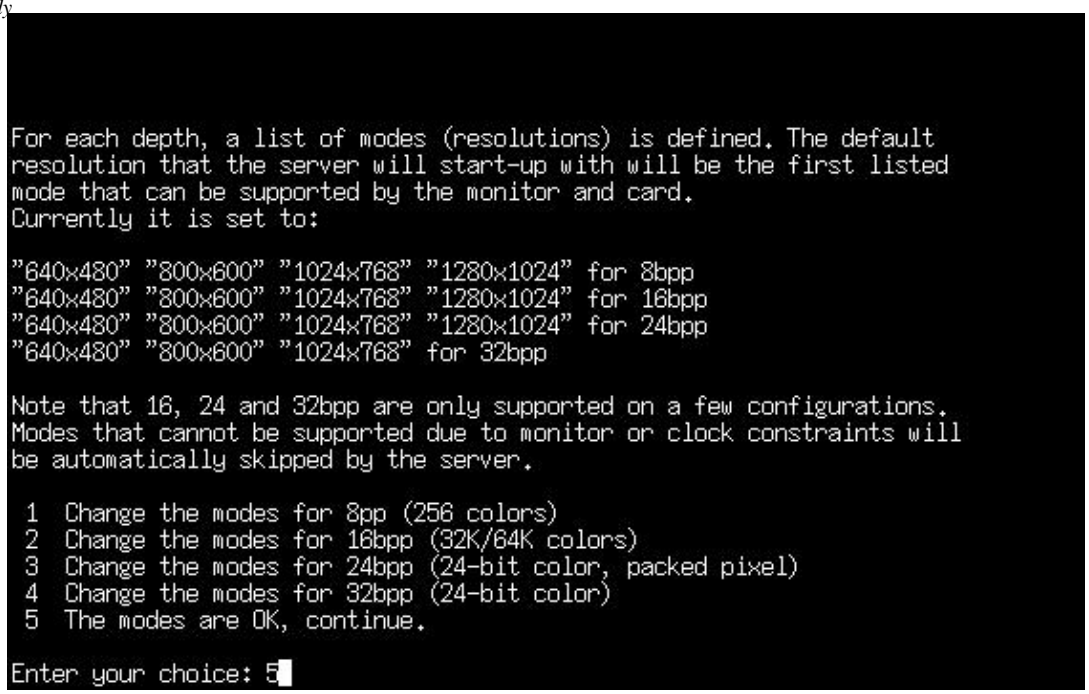
Clocks line

Dalš í obrazovka plná textu vysvětluje, co to clocks line je. Jak je tu řečeno, nebudete to na větš ině moderních konfigurací potřebovat. Potom se vás to zeptá, jestli by se měly hodiny testovat. Rovněž vám to řekne, jestli karta otestování potřebuje, nebo ne. V případě oné ATI karty **xf86config** říká:

```
   The card definition says to NOT probe clocks.
```

Pokud to říká něco takovéhoto, odpvězte "n" na otázku o testování karty na hodiny. Velmi staré grafické karty potřebují být testovány. **xf86config** vám řekne, co je potřeba udělat.

Video módy



Nyní přichází čas vybrat video módy, které váš X server bude používat. Uvidíte čtyři různé hloubky barev – 8bpp, 16bpp, 24bpp a 32bpp. Každá bude mít seznam různých videomódů, které mohou být provozovány při dané barevné hloubce. Když startujete X–ka, začne se s defaultní barevnou hloubkou a prvním rozlišením vypsaným u dané hloubky. Pokud chcete startovat X–ka defaultně v jiném rozlišení, máte to teď možnost nastavit.

Když je pořadí videomódů v pořádku, můžete vybrat "OK" a pokračovat v procesu nastavování. V opačném případě vyberte barevnou hloubku, kterou chcete změnit. Například předpokládejme, že vám byly nabídnuty následující možnosti:

```
"640x480" "800x600" "1024x768" "1280x1024" for 8bpp
"640x480" "800x600" "1024x768" "1280x1024" for 16bpp
"640x480" "800x600" "1024x768" "1280x1024" for 24bpp
"640x480" "800x600" "1024x768" for 32bpp
```

Chcete–li startovat X–ka defaultně v jiném rozlišení, vyberte nejdřív barevnou hloubku, kterou změnit. Potom se řiďte pokyny, které vám dává **xf86config**. Vyzve vás k zadání číslic, které odpovídají pořadí oněch rozlišení. Pokud byste třeba chěli jen jednoduše otočit pořadí rozlišení, mohli byste odpovědět:

```
Which modes? 5432
```

Rovněž máte možnost nějaké rozlišení vynechat. Pokud vaše videokarta nemůže běžet na 1280x1024, není důvod proč ho tam mít. Pak byste odpověděli:

```
Which modes? 432
```

Po vybrání módů a barevné hlouby budete dotázáni, jestli chcete mít virtuální obrazovku větší, než fyzickou. Virtuální obrazovka je obrazovka, která je větší než samotný monitor. Když pohybujete myší po virtuální obrazovce, ta bude rolovat o trochu dřív, než dojedete ke kraji. To vám dává možnost mít na monitoru více oken. Jenže protože nebudete moci vidět vše současně, může e vás virtuální obrazovka trochu zlobit. Je to jistě hezká věc na hraní, takže ji možná budete chtít vyzkoušet.

Pak se dostanete zpět do seznamu videomódů. Po změnění videomódů u 24bpp barevné hloubky by to mělo vypadat nějak takhle:

```
"640x480" "800x600" "1024x768" "1280x1024" for 8bpp
"640x480" "800x600" "1024x768" "1280x1024" for 16bpp
"1280x1024" "1024x768" "800x600" "640x480" for 24bpp
"640x480" "800x600" "1024x768" for 32bpp
```

Pokračujete ve změnách videomódu, dokud s tím nebudete spokojeni. Až budete s touto částí hotovi, zadejte "OK" a bude se pokračovat dál.

Zapsat konfigurační soubor

V tuto chvíli je nastavování X úplné.**xf86config** se vás zeptá, jestli má zapsat konfigurační soubor do `/etc/XF86Config`. Pokud chcete být schopni provozovat X–ka, měli byste na tuto otázku odpovědět "y". Do tohoto souboru se budou X–ka dívat na svoji konfiguraci.

Pokud jste odpověděli správně na všechny otázky a máte nainstalovaný balíček s X serverem, měli byste být schopni spustit X–ka. Takto:

```
$ startx
```

Pokud jste si nainstalovali KDE, nebo GNOME, mělo by teď naběhnout. Jinak možná budete potřebovat spoustit **xwmconfig** a vybrat si správce oken (window manager), který chcete, aby se používal jako výchozí. Správci oken budou popsáni dále v této kapitole. **xwmconfig** nastavuje defaultního okenního správce pouze pro uživatele, který jej spustil. Pokud na vašem systému pracuje více uživatelů, každý si bude muset vybrat svůj vlastní okenní

manaž er.

Je několik zvláš tních klávesových kombinací, které se při práci v X může hodit. Například, pokud potřebujete ukončit X–ka v určitém bodě a nemůž ete je zavřít korektním způsobem, je tu kobninace kláves pro násilné ukončení. **control–alt–backspace** zabije X–ka a smete vás zpět do příkazové řádky.
Do příkazové řádky se může t přepnout i za běhu X, když stisknete kombinaci kláves **CTRL–ALT–Fx** (x=číslo terminálu), což je obdoba přepínání mezi virtuálními terminály v konzolovém rež imu. Běž ící X jsou umístěna na terminálu č. 7, takž e zpět se do nich z konzole dostanete stiskem **alt–F7**.
A nakonec: Může te měnit videomódy za běhu X. **CTRL–ALT–+** (+ na num.klávesnici) přepne do vyš š ího rozliš ení, zatímco totéž s mínusem přepne do niž š ího rozliš ení.

Předchozí Shrnutí | Výchozí Nahoru | Dalš í XF86Setup

Kapitola 7. Bootování

Obsah

Proces bootování linuxového systému může být někdy snadný, někdy komplikovaný. Mnoho uživatelů si nainstaluje Slackware na svůj počítač a je to. Jenom ho zapnou a můžou ho používat. Jiní zas musí používat pro jisté úkoly jiný operační systém, takže potřebují mít oba operační systémy na svém stroji k dispozici.

Tato část knihy popisuje používání LILO a Loadlin, dva bootovače, které jsou součástí Slackware. Rovněž vysvětluje některé typické scénáře duálního bootování a jak si to můžete nastavit.

LILO

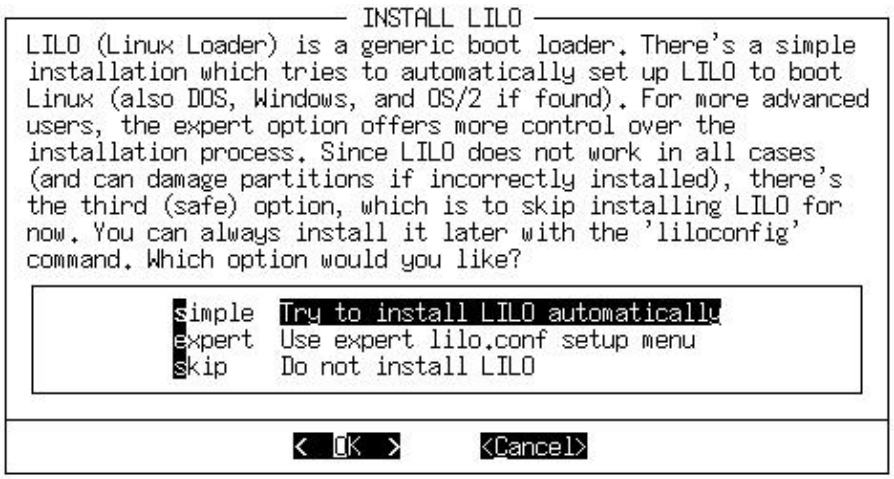
Linux Loader, neboli LILO (zavaděč Linuxu) je nejpopulárnějším zavaděčem (booter) používaným na linuxových systémech. Dobře se nastavuje a může být snadno používán k zavádění i jiných operačních systémů.

Slackware Linux přichází s nabídkově ovládanou konfigurační utilitou **liloconfig**. Tento program je prvně spouštěn v průběhu procesu instalace programem setup, ale můžete jej vyvolat i později příkazem **liloconfig** z příkazové řádky.

LILO čte svá nastavení ze souboru `/etc/lilo.conf`(5). Ten není načítán při každém bootování, ale je čten při každé instalaci LILO. LILO musí být reinstalováno do boot sektoru pokaždé, když provedete změny v nastaveních. **liloconfig** vám pomůže kofigurační soubor vytvořit tak, abyste mohli naistalovat LILO do vašeho systému. Dáváte–li přednost ruční editaci `/etc/lilo.conf`, pak přeinstalaci LILO spustíte příkazem **/sbin/lilo**.

Když poprvé spustíte **liloconfig**, bude to vypadat takto:

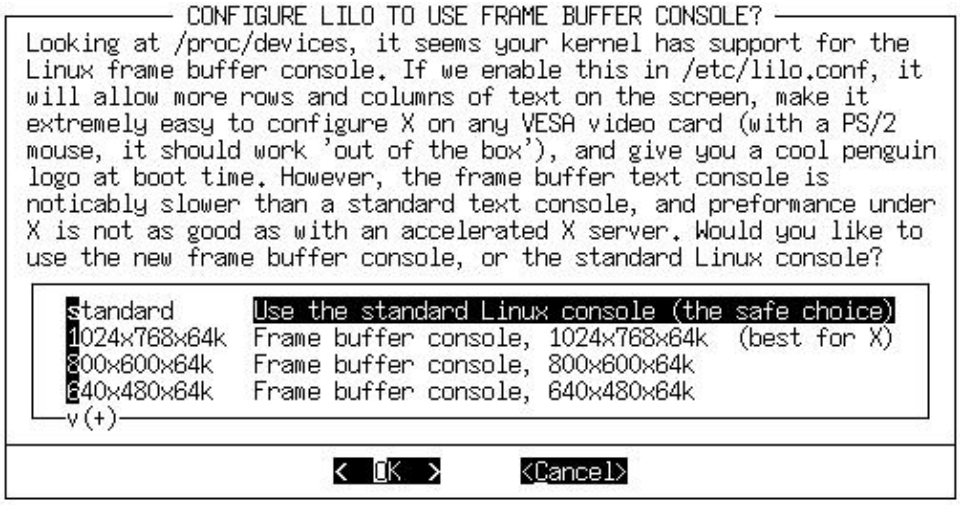
Obrázek 7–1. Úvodní obrazovka liloconfig.



Pokud nastavujete LILO poprvé, měli byste zvolit možnost "simple". Pokud ne, můžete shledat možnost "expert" jako rychlejší, zvláště jste–li dobře obeznámeni s LILO a Linuxem. Vybrání "simple" zahájí konfiguraci LILO.

Pokud máte v jádru zakompilovanou podporu pro frame buffer, pak se vás **liloconfig** bude ptát, jaké rozlišení zobrazení budete chtít používat. To je to rozlišení, které je rovněž používáno serverem frame bufferu systému XFree86. Nechcete–li provozovat konzoli ve zvláštním video módu, výběr "normal" ponechá standardní textový mód 80x25.

Obrázek 7–2. Liloconfig se ptá, jaký video mód se má použít pro frame buffer.



Další část konfigurace LILO spočívá v určení, kam budete chtít LILO nainstalovat. To je asi nejdůležitější krok. Následující seznam vysvětluje jednotlivá instalační umístění:

Root
: Tato volba nainstaluje LILO na začátek linuxového root oddílu. To je nejbezpečnější volba, máte–li na vašem počítači i jiné operační systémy. Zabezpečí, že zavaděče těchto systémů nebudou přepsány. Nevýhodou této možnosti je, že LILO odsud může bootovat jedině v případě, je–li linuxová jednotka první jednotkou ve vašem systému.

Floppy (disketa)
: Tato metoda je ještě bezpečnější než předchozí. Vytváří bootovací disketu, kterou můžete používat k bootování vašeho linuxového systému. To drží váš zavaděč zcela mimo harddisk. Z této diskety bootujte jenom, když chcete zavést Slackware.

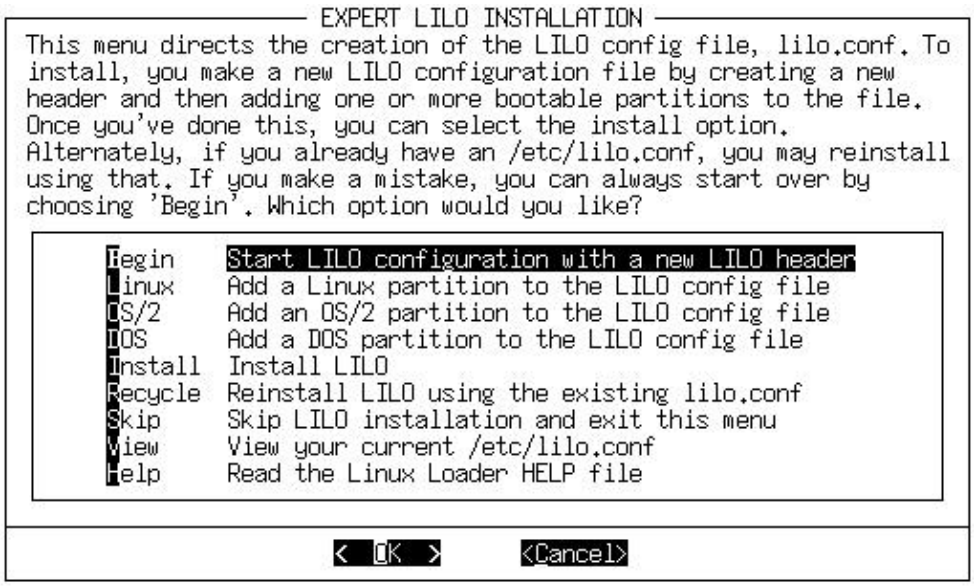
MBR
: Tuto možnost vyberete v případě, že je buď Slackware jediným operačním systémem ve vašem počítači, nebo když budete používat LILO k vybírání z více operačních systémů ve vašem počítači.

Tato volba přepíše jakýkoliv zavaděč, který máte v MBR.

Po výběru umístění LILO, zapíše **liloconfig** konfigurační soubor a nainstaluje LILO. A je to.

Když si vyberete mód "expert", obdržíte speciální menu. Toto menu vám umožňuje prohlížet soubor `/etc/lilo.conf`, přidávat další operační systémy do bootovací nabídky a zadat do LILO speciální parametry, které budou předávány jádru během bootování. Menu "expert" vypadá takto:

Obrázek 7–3. liloconfig – mód expert .



Ať už je konfigurace vašeho systému jakákoliv, nastavení funkčního bootovacího zavaděče je snadné. **liloconfig** dělá z jeho nastavení jasnou věc. Nicméně mohou nastat situace, kdy LILO na vašem systému nebude fungovat. Naštěstí tu jsou ještě jiné možnosti.

Kapitola 8. Shell

Obsah

V grafickém prostředí je uživatelské rozhraní vytvářeno programem, který vykresluje okna, posuvníky, nabídky, atd. V prostředí příkazové řádky vytváří uživatelské rozhraní "shell", který interpretuje příkazy a obecně činí věci použ ívatelnými. Bezprostředně po přihláš ení do systému (což je popsáno dále v této kapitole) se uživatel dostane do shellu a může se věnovat svým záležitostem. Tato kapitola slouží jako úvod do shellu, a to do nejrozš ířenějš ího shellu mezi uživateli Linuxu−− Bourne Again Shellu (bash). Detailnějš í informace o čemkoliv z této kapitoly hledejte v manuálové stránce **bash**(1).

Uživatelé

Přihlašování

Takže jste nabootovali a díváte se na něco co vypadá asi takhle:

```
Welcome to Linux 2.2.14
 darkstar login:
```

Hmm.. nikdo nic neříká o přihlaš ování (login). A co je to to darkstar (pp: V angličtině "temná hvězda")? Nebojte se, nezaž ehli jste náhodně hyperprostorový comm−link do Říš ského umělého měsíce. (Obávám se, ž e hyperprostorový comm−linkový protokol není v současnosti linuxovým jádrem podporován – pp: Pro zcela neznalé – tohle byl vtip.) Ne, darkstar je pouze jméno jednoho z naš ich počítačů a jeho jméno se objevuje standardně. Pokud jste během instalace zadali nějaké jméno pro váš počítač, mělo by se tu objevit místo "darkstar".

Pokud jde o "login"... Jestliž e se přihlaš ujete (logujete) poprvé, musíte se přihlásit jako **`root`**. Budete pož ádáni o zadání hesla. Pokud jste nějaké zadali během instalace, tak na to se vás to teď ptá. Pokud jste ž ádné nezadávali, jen stiskněte enter. A je to−− jste uvnitř!

Root: Superuživatel

Tak jo, ale kdo nebo *co* je "root"? A co dělá jeho účet na *mém* systému?

Takže: Ve světě Unixu a podobných operačních systémů (jako je Linux), existují uživatelé ale taky uživatelé. Dostanem se do toho detailně později, ale důležitá věc k vědění teď je, ž e root je uživatel nad vš emi uživateli; root je vš emohoucí a vš evědoucí a *nikdo* nesmí roota neposlechnout. Prostě se to nesmí. Root je ten, koho zveme "superuživatel", a po právu. A nejlepš í ze vš eho je, ž e root jste *vy*.

Bomba, co?

Nejste−li si jisti: Ano, je to opravdu bomba. Chyták je v tom, ž e rootovi je umož něno rozflákat cokoliv, na co si troufne. Mož ná byste teď měli přeskočit do 12. kapitoly a podívat se, jak se přidávají uživatelé; potom se přihlásit jako normální uživatel a pracovat tak. Tradiční moudrost praví, ž e bychom se měli stávat superuživateli jen v nezbytných případech, abychom minimalizovali riziko, ž e něco poš kodíme náhodně.

Mimochodem: Pokud se rozhodnete stát rootem v situaci kdy jste přihláš eni jako normální uživatel, můž ete použ ít příkaz **su**(1). Ten se vás zeptá na rootovo heslo a můž ete pracovat jako root dokud se zase neodhlásíte příkazem **exit** nebo **logout** (pp: nebo ctrl+d). Příkz **su**, nabízí mož nost stát se jakýmkoliv jiným uživatelem, pokud znáte jeho heslo. Například **su franta** vás přihlásí jako uživatele "franta".

Kapitola 9. Struktura souborového systému

Obsah

Již jsme se bavili o struktuře adresářů v Slackware Linuxu. Umíte najít soubory a adresáře které potřebujete. Ale souborový systém je něčím víc, než jen strukturou adresářů.

Linux je více–uživatelský operační systém. Každý aspekt tohoto systému je více–uživatelský, i souborový systém. Systém ukládá informace o tom kdo vlastní soubor a kdo jej smí číst. Jsou tu i další úkoly souborového systému, jako odkazy a NFS připojování. Tato část knihy to vysvětluje, stejně tak jako více–uživatelský aspekt souborového systému.

Vlastnictví

Souborový systém ukládá informace o vlastnictví pro každý soubor a adresář v systému. Tj. který uživatel a skupina vlastní ten který soubor. Nejsnazší způsob jak zobrazit tyto informace je použít příkaz **ls**:

```
$ ls -l /usr/bin/wc
 -rwxr-xr-x   1 root     bin    7368 Jul 30  1999 /usr/bin/wc
```

Zajímá nás třetí a čtvrtý sloupec. Ty obsahují jméno uživatele a skupiny, kteří vlastní tento soubor. Vidíme, že tento soubor vlastní uživatel "root" a skupina "bin".

Vlastníky můžeme měnit příkazy **chown**(1) (značí "change owner" – změň vlastníka) a **chgrp**(1) (značí "change group" – změň skupinu). Abychom změnili vlastníka souboru na "daemon", použijeme **chown**:

```
# chown daemon /usr/bin/wc
```

Ke změně skupiny vlastníků na "root" použijeme **chgrp**:

```
# chgrp root /usr/bin/wc
```

Pomocí **chown** můžeme změnit i vlastníka i skupinu:

```
# chown daemon.root /usr/bin/wc
```

Vlastnictví souborů je velmi důležitou částí práce s linuxovým systémem, a to i když jste jen uživatelem. Občas potřebujete upřesnit vlastnictví souborů a nódů zařízení.

Kapitola 10. Manipulace se soubory a adresáři

Obsah

Slackware Linux se snaží být Unixoidním, jak jen to je možné. Tradičně jsou Unixové operační systémy orientované na práci v příkazové řádce. V Slackwaru je k dispozici i grafické uživatelské rozhraní, ale příkazová řádka stále zůstává hlavním místem pro ovládání systému. Z toho důvodu je důležité porozumnět několika základním příkazům pro správu souborů.

Následující sekce popisují obecné příkazy pro správu souborů a vysvětlují jejich použití. Existuje řada dalších příkazů; ty zde uvedené by vám měly pomoci do začátku. Rovněž tu jsou tyto příkazy popsány zestručněně. Více podrobností o nich najdete v příslušných manuálových stránkách.

ls

(list) Tento příkaz vypíše soubory v adresáři. Uživatelé Windows a DOSu si všimnou, že se podobá příkazu **dir**. Sám o sobě **ls**(1) vypíše soubory ležící v aktuálním adresáři. Chcete–li vidět, co je ve vašem kořenovém adresáři, musíte zadat příkaz takto:

```
$ ls /
bin   cdr    dev  home  lost+found  proc  sbin   tmp  var
boot  cdrom  etc  lib   mnt         root  suncd  usr  vmlinuz
```

Bohužel z tohto výpisu nepoznáte co je soubor a co adresář. Můžete k příkazu **ls** zadat paramtery, které do výpisu přidají identifikátory:

```
$ ls -FC
  bin/   cdr/    dev/  home/  lost+found/  proc/  sbin/   tmp/  var/
  boot/  cdrom/  etc/  lib/   mnt/         root/  suncd/  usr/  vmlinuz
```

Adresáře jsou označené lomítkem za jménem, spustitelné soubory tam mají hvězdičku, symbolické odkazy zavináč,... "normální" soubory tam nemají nic.

Příkaz **ls** rovněž umožňuje získat další údaje o souborech. Například pro zobrazení oprávnění, počtu pevných odkazů, vlastníka, skupiny, velikosti a data vytvoření si můžete vyžádat dlouhou (long) verzi výpisu:

```
$ ls -l
  drwxr-xr-x   2 root     bin          4096 May  7  1994 bin/
  drwxr-xr-x   2 root     root         4096 Feb 24 03:55 boot/
  drwxr-xr-x   2 root     root         4096 Feb 18 01:10 cdr/
  drwxr-xr-x  14 root     root         6144 Oct 23 18:37 cdrom/
  drwxr-xr-x   4 root     root        28672 Mar  5 18:01 dev/
  drwxr-xr-x  10 root     root         4096 Mar  8 03:32 etc/
  drwxr-xr-x   8 root     root         4096 Mar  8 03:31 home/
  drwxr-xr-x   3 root     root         4096 Jan 23 21:29 lib/
  drwxr-xr-x   2 root     root        16384 Nov  1 08:53 lost+found/
  drwxr-xr-x   2 root     root         4096 Oct  6  1997 mnt/
  dr-xr-xr-x  62 root     root            0 Mar  4 15:32 proc/
  drwxr-x---x  12 root     root        4096 Feb 26 02:06 root/
  drwxr-xr-x   2 root     bin          4096 Feb 17 02:02 sbin/
  drwxr-xr-x   5 root     root         2048 Oct 25 10:51 suncd/
  drwxrwxrwt   4 root     root       487424 Mar  7 20:42 tmp/
  drwxr-xr-x  21 root     root         4096 Aug 24  1999 usr/
  drwxr-xr-x  18 root     root         4096 Mar  8 03:32 var/
  -rw-r---r---   1 root     root       461907 Feb 22 20:04 vmlinuz
```

Kdybyste chtěli, aby se vypsaly i skryté soubory, přidejte prametr –a nebo –A:

```
$ ls -a
  .              bin   cdrom  home        mnt   sbin   usr
  ..             boot  dev    lib         proc  suncd  var
  .pwrchute_tmp  cdr   etc    lost+found  root  tmp    vmlinuz
```

"Skryté" jsou ty soubory, jejichž název začíná tečkou. Také jim říkáme tečkové soubory (dot files).

Příkaz **ls** má ještě spoustu dalších voleb, které najdete v manuálové stránce (man ls). Nezapomeňte taky, že jednotlivé volby můžete navzájem kombinovat.

Kapitola 11. Správa proceů

Obsah

Každý běžící program se nazývá proces. Tyto procesy zahrnují vše od takových věcí jako je X Window Systém až po systémové programy (démony), které se spouštějí během startu počítače. Každý proces běží jako nějaký konkrétní uživatel. Procesy spouštěné během bootu (zavádění systému) obvykle běží jako root nebo nobody. Procesy, které spouštíte vy, poběží jako vy. Procesy spuštné jiným uživatelem poběží jako tento uživatel.

Máte vládu nad procesy, které jste sputili vy. Navíc root má vládu úplně nad všemi procesy v systému, včetně těch, které spustili jiní uživatelé. Procesy mohou být ovládány a monitorovány pomocí několika programů a rovněž několika shellovými přikazy.

Backgrounding

Programy spouštěné z příkazové řádky nastartují na popředí (foreground). To vám umožňuje vidět veškerý výstup těchto programů a komunikovat s nimi. Nicméně existují případy, kdy byste dali přednost spuštění programu bez toho, aby zabral terminál. Tomu říkáme spouštění programů na pozadí (background). Je několik způsobů, jak toho dosáhnout.

První způsob jak dostat proces na pozadí je přidat znak ampersand na příkazovou řádku, kterou spouštíte program. Například předpokládejme, že byste chtěli použít příkazo–řádkový přehrávač mp3 **mpg321**, aby přehrával adresář plný mp–trojek, ale potřebovali byste na tom samém terminálu dělat něco jiného. Následující příkazový řádek spustí mpg321 na pozadí:

```
$ mpg321 *.mp3 &
```

Program se rozběhne jako obvykle, ale vrátí se vám prompt na monitor.

Jiný způsob jak dostat proces na pozadí je udělat to, když už program běží. Nejprve spusťte nějaký program. Když běží, stiskněte **control+z**. To ten proces pozastaví. Okamžitě zastaví jeho běh, ale může být kdykoliv znovu rozběhnut. Jakmile je proces pozastaven, vrátí se vám na monitor prompt. Pak můžete rozběhnout pozastavený proces na pozadí tak, že napíšete:

```
$ bg
```

Nyní běží předtím pozastavený proces na pozadí.

Kapitola 12. Základní administrace systému

Obsah

Jste administrátorem nějakých počítačů a máte tam práva roota. Možná je to váš desktop s jedním či dvěma uživateli, možná je to velký server, kde jich je několik set. Bez ohledu na to budete potřebovat vědět, jak uživatele spravovat a jak bezpečně ukončovat práci systému. Obé je zdánlivě snadné, ale věnujme se tomu chvíli. Na závěr si řekneme pár myšlenek, na kterých pracuje systém hesel.

Uživatelé a skupiny

Dodávané skripty

Nejjednodušším způsob jak spravovat uživatele a skupiny je používat dodávané skripty a programy. Slackawre obsahuje programy **adduser**, **userdel**(8), **chfn**(1), **chsh**(1) a **passwd**(1), které pečují o uživatele. Dále programy **groupadd**(8), **groupdel**(8) a **groupmod**(8), které se věnují skupinám uživatelů. S výjimkou **chfn**, **chsh** a **passwd** to jsou programy, které může spouštět pouze root a proto jsou umístěny v `/usr/sbin`. **chfn**, **chsh** a **passwd** může spouštět kdokoliv a jsou umístěny v `/usr/bin`.

Přidání uživatele

Uživatele přidáváte do systému pomocí programu **adduser**. Projďeme si teď celou proceduru a proberme si všechny otázky, které nám budou kladeny. Také si stručně vysvětlíme co znamenají. Defaultní odpověď je zobrazena v hranatých závorkách a může být použita téměř u všech otázek – pokud samozřejmě nechcete něco zadat jinak.

```
# adduser
 Login name for new user (8 characters or less) []: jellyd
```

Přihlašovací jméno pro nového uživatele (8 či méně znaků) [ ]:
Tohle je jméno, které bude uživatel používat při přihlašování do systému. Může být dlouhé nejvýše 8 znaků, protože tak to přihlašovací utility očekávají. Obecně byste měli používat jen znaky bez shiftu (malá písmena ap), tedy pokud se vám nechce na obtížných místech přepínat na písmena velká.

```
User id for jellyd [ defaults to next available]:
```

Uživatelské číslo pro uživatele jellyd [defaultně následující volné]:
Skutečné vlastnictví pod Linuxem je určováno pomocí UID (user ID). To je jedinečné číslo, které je přiděleno každému uživateli – ve Slackwaru počínaje 1000. UID pro nového uživatele můžete zadat sami, nebo můžete nechat na programu **adduser**, aby mu sám přidělil první volné číslo.

```
Initial group for jellyd [users]:
```

Výchozí skupina pro uživatele jellyd [users]:
Všichni uživatelé jsou defaultně zařazováni do skupiny "users" (uživatelé). Můžete zařadit uživatele do jiné skupiny, ale nedoporučuje se to.

```
Additional groups for jellyd (seperated with commas, no spaces) []:
```

Další skupiny, kam zařadit uživatele jellyd (oddělujte je čárakmi bez mezer) [ ]:
Tato otázka vám umožňue zařadit nového uživatele ještě do dalších skupin. Každý uživatel může být členem několika skupin najednou. To se hodí, když třeba máte zřízeny skupiny pro takové záležitosti jako je modifikace souborů webových stránek, hraní her ap.

```
jellyd's home directory [/home/jellyd]:
```

domovský adresář pro uživatele jellyd [/home/jellyd]:
Domovský adresář uživatele je defaultně umístěn v adresáři `/home` a je mu přidělen název shodný s jeho přihlašovacím jménem. Jestliže provozujete rozsáhlý systém je možné, že máte domovské adresáře zařazené podle jiného schématu. Takže tato otázka vám umožnuje zadat místo, kde uživatelův domovský adresář bude doopravdy. Rovněž můžete zablokovat jakýkoliv účet tak, že změníte uživatelův domovský adresář na něco jako `/bin/false`, i když tohle je metoda, která se moc nedoporučuje.

```
jellyd's shell [/bin/bash]:
```

shell, který bude jellyd používat [/bin/bash]:
Ve Slackwaru se jako stadnardní používá shell **bash** a pro většinu lidí je skutečně vhodný. Pokud nový uživatel přichází z Unix–ového prostředí, je už možná sžitý s jiným shellem. Můžete mu změnit shell nyní, nebo si ho pak může změnit sám dodatečně pomocí příkazu **chsh**

```
jellyd's account expiry date (YYYY-MM-DD) []:
```

Jellydho účet vyprší dnem (YYYY–MM–DD) [ ]:
U účtu můžete nastavit, do kdy bude platit. Defaultně se toto datum nenastavuje (účet je navždy). Můžete to změnit, chcete–li. Tato volba se může hodit lidem fungujícím jako placená veřejná služba (třeba ISP). Ti pak mohou nechat účet "propadnout", když do stanoveného data nepřijde platba na další období.

```
OK, I'm about to make a new account. Here's
 what you entered so far:

 New login name: jellyd
 New UID: [Next available]
 Initial group: users
 Additional groups: [none]
 Home directory: /home/jellyd
```

```
  Shell: /bin/bash
  Expiry date: [no expiration]

  This is it... if you want to bail out, hit Control-C.
  Otherwise, press ENTER to go ahead and make the account.
```

Tady máte souhn informací, které jste zadali pro nový účet a je vám dána poslední mož nost je zkontrolovat a buď schválit, nebo odmítnout. Pokud jste zadali něco nesprávně, budete teď muset stisknout Control–C a začít znovu. Jinak stiskněte Enter a účet bude vytvořen:

```
 Making new account...

  Changing the user information for jellyd
  Enter the new value, or press return for the default
          Full Name []: Jeremy
          Room Number []: Smith 130
          Work Phone []:
          Home Phone []:
          Other:
```

Po vypsání informace, ž e účet byl zřízen, je vám polož eno ješ tě několik nepovinných otázek. Nemusíte zadávat nic, nechcete–li a už ivatel si pak můž e vš e nastavit pomocí příkazu **chfn** dodatečně. Nicméně se můž e hodit zadat aspoň plné jméno a telefonní čísla, čistě pro případ, ž e se s danou osobou budete chctít setkat osobně.

Dál budete ž ádáni o zadání hesla:

```
 Changing password for jellyd
  Enter the new password (minimum of 5, maximum of 127 characters)
  Please use a combination of upper and lower case letters and numbers.
  New password:
  Re-enter new password:
  Password changed.
  Done...
```

Heslo pro nového už ivatele budete muset zadat. Obecně, není–li nový už ivatel fyzicky přítomen v okamž iku kady mu zakládáte účet, vlož te nějaké standardní heslo a sdělte už ivateli, aby si ho dodatečně změnil na něco více bezpečného.

> **Vybíráme heslo**
>
> Mít bezpečné heslo je prvořadá obrana proti cracknutí. Něměli byste chtít mít jednoduš e uhodnutelné heslo, protož e to velmi usnadňuje případnému "zájemci" průnik do vaš eho systému. Ideálním a bezpečným heslem je náhodný řetězec znaků, obsahující velká a malá písmena, čísla a náhodné znaky. Pouze si zapamatujte, ž e znak tab (tabulátor) není vhodný, protož e na různých systémech můž e odesílat jinou sekvenci znaků.
>
> Obecně se řiďte zdravým rozumem: Nezadávejte jako heslo něčí datum narození, obecnou frázi, něco, co se nalézá na vaš em stole, nebo vůbec cokoliv, co lze snadno spojovat s vaš í osobou. "secure1" je rovněž š patné.

Odstranění už ivatele

Odstranit už ivatele není vůbec těž ké. Pouze spusť te **userdel** se jménem účtu, který chcete zruš it. Nejdřív se ale přesvěčte, zda onen už ivatel není právě přihláš en v systému, a ž e ž ádný proces neběž í pod jeho účtem. Také si zapamatujte, ž e jakmile už ivatele smaž ete, on opravdu zmizí.

```
  # userdel jellyd
```

Takž e takhle odstraníte nepříjemného chlapíka jména "jellyd" z vaš eho systému. Prima výkop! :) Tohle vymaž e už ivatele ze souborů `/etc/passwd` a `/etc/group`, ale neodstraní to už ivatelův domovský adresář. Pokud chcete jedním vrzem smáznout i ten, udělejte to takhle:

```
 # userdel -r jellyd
```

Dočasné zablokování nějakého účtu bude popsáno v sekci *Měníme heslo*, protož e to zahrnuje změnu už ivatelova hesla. Změna informací o účtu je popsána v sekci *Měníme heslo* a v sekci *Měníme informace o už ivateli*.

Přidání a odstranění skupiny

Programy pro přidání a vyjmutí skupin jsou velmi jednoduché. **groupadd** pouze přidá dalš í skupinu do souboru `/etc/group` s jedinečným ID skupiny, zatímco **groupdel** zadanou skupinu odstraní. Pro přidávání už ivatelů do určitých skupin pak můž ete ručně editovat soubor `/etc/group`.

Skupinu vytvoříte takto:

```
  # groupadd cvs
```

A vyjmete jí takto:

```
  # groupdel cvs
```

Ručně

Pochopitelně je mož né přidávat, měnit a odstraňovat už ivatele a skupiny ručně. Po projití tohoto postupu nejspíš zjistíte, ž e je mnohem výhodnějš í použ ívat výš e popsané programy a skripty, si myslim.

Nejdříve přidáme nového už ivatele do souborů `/etc/passwd`(5), `/etc/shadow`(5) a `/etc/group`(5). Soubor `passwd` slouž í k uchovávání některých informací o už ivatelích, ale ne (poněkud nelogicky) jejich hesel. Tento soubor musí být přístupný pro čtení komukoliv. Jenž e vy nechcete, aby kódovaná hesla byla čitelná kýmkoliv na světě, protož e to by dávalo rádoby–crackerům dobrý počinek do začátku. Takž e kódovaná hesla jsou

ukládána do souboru /etc/shadow, který je čitelný pouze pro roota a do souboru passwd se místo hesla ukádá jen písmeno "x". Soubor /etc/group obsahuje seznam vš ech skupin a jejich členů.

Takž e si pojďme vyzkouš et přidání už ivatele do /etc/passwd. Typický řádek v souboru passwd vypadá takto:

```
chris:x:1000:100:Chris Lumens,Room 2,,:/home/chris:/bin/bash
```

Kaž dý řádek je záznamem jedné osoby a pole na kaž dé řádce jsou oddělena dvojtečkou. Těmi poli jsou: Přihlaš ovací jméno, zakódované heslo (ve Slackwaru tu je "x", protož e na hesla použ íváme "shadow"), už ivatelovo ID, ID základní skupiny, volitelné informace pro "finger" oddělené čárkami, domovský adresář a shell. Takž e na konec souboru přidejte řádek s potřebnými informacemi a je to.

Přesvědčte se, ž e heslo je **x**, ž e už ivatelovo ID je jedinečné, ž e jste mu dali číslo skupiny 100 (skupina "users" ve Slackwaru), a ž e mají platný shell.

Dále potřebujeme přidat záznam do souboru /etc/shadow, kde se uchovávají hesla. Typický záznam v tomto souboru vypadá takto:

```
chris:$1$w9bsw/N9$UWLr2bRER6YyBS.CAEp7R.:11055:0:99999:7:::
```

Opět co řádka, to jedna osoba a údaje jsou odděleny dvojtečkou. Údaji jsou:

- přihlaš ovací jméno,
- zakódované heslo,
- počet dní od "počátku světa" (1.1.1970) kdy bylo heslo naposledy měněno,
- za kolik dní smí být heslo změněno,
- za kolik dní se musí heslo měnit,
- kolik dní před vyprš ením platnosti hesla má být už ivatel upozorněn,
- po kolika dnech po vyprš ení platnosti hesla bude účet zablokován,
- po kolika dnech od "počátku světa" bude účet zablokován
- a jedno rezervované pole.

Jak vidíte, větš ina údajů se týká doby platnosti účtu. Pokud nepouž íváte informace o platnosti, pak vyplníte pouze pár polí. Pokud ano, pak budete muste provést pár propočtů a rozhodování dříve, než se pustíte do vyplňování. Pro naš eho nového už ivatele zadejte na místo hesla jen nějaké náhodné "smetí". Co takové heslo bude znamenat se nestarejte, protož e už za minutku ho stejně změníte. Pozor jen na ten jediný znak, který tam být nesmí: dvojtečka. Následující pole "počet dní od poslední změny hesla" nechte prázdné. Do dalš ích polí zadejte **0**, **99999** a **7** tak, jak vidíte v příkladu řádku výš e a zbývající políčka nechte prázdná.

For those of you who see my encrypted password above and think you've got a leg up on breaking into my system, go right ahead. If you can crack that password, you'll know the password to a firewalled test system. Now that's useful :)

Jelikož každý je defaultně členem skupiny "users", nemusíte do ní nového už ivatele přidávat. Chcete–li vytvořit novou skupinu, nebo přidat nového už ivatele do stávajících skupin, budete muset upravit soubor /etc/group. Zde máte typický záznam:

```
cvs::102:chris,logan,david,root
```

Jednotlivými poli jsou: Jméno skupiny, heslo skupiny, ID skupiny a členové skupiny (seznam oddělený čárkou bez mezer). Vytvoření nové skupiny je snadná zálež itost spočívající v přidání nového řádku s jedinečným ID skupiny a seznamem osob, které mají být jejími členy. Pokud je v čase přidání už ivatele do skupiny tento už ivatel přihláš en v systému, bude se muset odhlásit a znovu přihlásit, aby se pro něj tato změna uplatnila.

Nyní použ ijte příkaz **passwd**, abyste novému už ivateli vytvořili heslo. Pak pomocí **mkdir** vytvořte novému už ivateli domovský adresář tak, jak jste uvedli v souboru /etc/passwd.

Pokud jste si do systému nainstalovali **sendmail**(8) pro doručování poš ty a aktivně mail použ íváte, potřebujete pro nového už ivatele vytvořit nový soubor v /var/spool/mail se správnými právy a vlastnictvím. Zde je příklad:

```
 # touch /var/spool/mail/jellyd
# chown jellyd.users /var/spool/mail/jellyd
# chmod 660 /var/spool/mail/jellyd
```

Tyto příkazy vytvoří novému už ivateli "jellyd" soubor pro ukládání poš ty a nastaví správně vlastnictví a přístupová práva.

Ruční odstranění už ivatele je jednoduchá zálež itost spočívající v odstranění vš eho, co jsme právě vytvořili. Odstraňte záznam o už ivateli z /etc/passwd, z /etc/shadow a z /etc/group. Odstraňte jeho jméno ze vš ech skupin v /etc/group, vyjměte jeho mail–spoolový soubor, pokud jej má a dle uváž ení smaž te i jeho domovský adresář.

Odstraňování skupin spočívá ve vymazání záznamu v /etc/group.

Měníme hesla

Program **passwd** mění hesla tak, ž e upravuje záznamy v /etc/shadow. Tento soubor uchovává v zakódovaném tvaru vš echna hesla systému. Abyste změnili svoje heslo, musíte zadat:

```
$ passwd
Changing password for chris
Old password:
Enter the new password (minumum of 5, maximum of 127 characters)
Please use a combination of upper and lower case letters and numbers.
New password:
```

Jak vidíte, program vás nejprve vyzve k zadání vaš eho starého hesla. Až ho budete psát, nebude se zobrazovat na obrazovce – stejně jako když se přihlaš ujete do systému. Pak vás vyzve k zadání nového hesla. Program **passwd** provádí na novém heslu několik kontrol a bude si stěž ovat, když při těchto kontrolách vaš e heslo neuspěje. Tato varování můž ete ignorovat, chcete–li. Nakonec budete vyzváni k opětovnému zapsání hesla – pro potvrzení.

Pokud jste root, můž ete rovněž měnit hesla ostatních už ivatelů:

```
# passwd ted
```

Poté budete muset absolvovat stejnou proceduru jako výše, s výjimkou toto, že nebudete muset zadávat stávající heslo. (Jedna z řady výhod, když jste rootem...)

Vyskytnou–li se ve vašem systému nějací "potížisté", můžete jim dočasně zablokovat účty. Později jim je zase můžete znovu zpřístupnit. Zablokování i odblokování účtu provedete pomocí příkazu **passwd**. Pro zablokování účtu udělejte jako root toto:

```
# passwd -l david
```

To změní Davidovo heslo na něco, co není možné dosáhnout jakýmkoliv kódováním čehokoliv (je vytvořen neplatný řetězec, začínající znakem ! – pozn.překl.). Heslo zplatníte zpět, a tím odblokujete uživateli účet tak, že zadáte toto:

```
# passwd -u david
```

Nyní je Davidův účet opět přístupný. Zablokování účtu se může hodit v případech, kdy uživatel nehraje podle vámi stanovených pravidel, nebo když vyexportoval obrovskou kopii **xeyes**(1) na váš X desktop.

Měníme informace o uživateli

Jsou dva údaje, které si uživatel může ve svém účtu kdykoliv změnit: Jeho shell a informace pro finger. Slackware pro tuto změnu používá příkazy **chsh** (change shell – změň shell) a **chfn** (change finger – změň finger).

Uživatel si může zvolit libovolný shell, který je uveden v souboru `/etc/shells`. Pro většinu lidí je **bash** akorát. Jiní mohou být zvyklí z dřívějška na shell, který se nacházel na jejich Unix–ovém systému v práci či škole a chtějí používat to, co už znají. Shell se mění použitím příkazu **chsh**:

```
$ chsh
 Password:
 Changing the login shell for chris
 Enter the new value, or press return for the default
      Login Shell [/bin/bash]:
```

Nejdříve zadáte své heslo a potom vložíte jméno žádaného shellu včetně jeho úplné cesty. Ujistěte se nejdříve, že je uveden v seznamu v souboru `/etc/shells`(5). Root může měnit shell pro ostatní uživatele tak, že zadá **chsh** se jménem uživatele coby parametrem.

Informace pro finger je nepovinná a může obsahovat vaše celé jméno, telefonní čísla a číslo místnosti. Změnit je můžete použitím **chfn**, a pokračovat stejnou procedurou, kterou jsme ukazovali při zakládání účtu. Jako obvykle, root smí měnit informace pro finger jakémukoliv uživateli.

Předchozí: Shrnutí | Výchozí: Nahoru | Další: Správné vypínání systému

Kapitola 13. Základní síťové příkazy

Obsah

Síť tvoří několik počítačů propojených dohromady. Síť může být jednoduchá, třeba jako pár počítačů propojených u vás doma nebo v kanceláři, nebo složitá jako rozsáhlá univerzitní síť, nebo přímo celý Internet. Je–li váš počítač částí sítě, pak máte do ostatních počítačů přístup buď přímo, nebo pomocí služeb jako mail nebo web.

Existuje velké množství síťových programů, které můžete používat. Některé jsou určené k diagnostikování sítě, abyste viděli, zda vše chno správně pracuje. Jiné (jako mailoví klienti nebo webové prohlížeče) můžete používat pro svou práci a pro kontakt s druhými lidmi.

ping

Program **ping**(8) posílá paket ICMP ECHO_REQUEST na zadaný hostitelský počítač. Pokud tento počítač odpovídá, vrátí se vám ICMP paket zpátky. K čemu to je? No, můžete "pingnout" libovolnou IP adresu, abyste věděli, jestli ten počítač běží. Pokud odpověď nepřijde, víte, že něco není v pořádku. Tady máte příklad rozhovoru mezi dvěma linuxovými uživateli:

> Franta: Míra to už má zase vypnutý.
>
> Tonda: Jseš si jistej?
>
> Franta: Bodejť. Zkusil jsem ho pingnout, ale nepřišla mi odpověď.

To dokládá, jak je **ping** užitečným příkazem pro každodenní využití. Poskytuje velice rychlou cestu jak zjistit, jestli je stroj zapnut a připojen do sítě. Základní syntaxe je:

```
$ ping <ip address or hostname>
```

Ping má samozřejmě několik voleb, které můžete zadat. Podrobnosti hledejte v manuálových stránkách **ping**(1).

Kapitola 14. Archivujeme soubory

Obsah

Slackware Linux nabízí několik programů, které můžete použít ke kompresi a archivování souborů. Tyto programy jsou obzvláště užitečné pro vytváření záloh a pro posílání kopií souborů mezi počítači prostřednictvím sítě. Tyto programy si poradí jak s Unixovými, tak i Windowsovými formáty archivů.

gzip

gzip(1) je kompresní program z projektu GNU. Pracuje tak, že zkomprimuje jeden zadaný soubor. Základní použití vypadá takto:

```
$ gzip infile
```

Výsledný soubor bude pojmenován `infile.gz` a bude obvykle menší, než onen vstupní soubor. Všimněte si, že `infile` je nahrazen `infile.gz`. To znamená, že `infile` už nebude existovat. Pouze jeho gzipovaná forma.

Obyčejné texty se zkomprimují výrazně. Naproti tomu obrázky ve formátu jpeg, soubory mp3 a další podobné se zkomprimují málo, pokud vůbec. Je to proto, že soubory těchto formátů už vlastně zkomprimované jsou.

Můžete rovněž vybalancovat optimalizaci velikosti komprimace, nebo její rychlosti. Maximální komprese dosáhnete takto:

```
$ gzip -9 infile
```

Takto bude komprimace trvat déle, ale výsledný soubor bude nejmenší, jaký umí **gzip** vyrobit. Zadáním nižších hodnot ve volbě v příkazové řádce dosáhnete kratšího času komprimace, ale komprese nebude taková.

Dekompresi gzipovaných programů můžete provést pomocí dvou příkazů, které jsou ale ve skutečnosti tím samým programem. Program **gzip** dekomprimuje jakýkoliv soubor s rozpoznanou příponou. Tou může být některá z těchto: `.gz`, `-gz`, `.z`, `-z`, `.Z`, nebo `-Z`. První metoda dekomprese je program **gunzip**(1), použitý takto:

```
$ gunzip infile.gz
```

To vytvoří dekomprimovanou verzi souboru `infile` v aktuálním adresáři a přípona .gz bude z názvu souboru odstraněna.

Druhým způsobem je použít opět **gzip**, ale s volbou **–**d:

```
$ gzip -d infile.gz
```

To způsobí přesně totéž, jako volání gunzip–u. Důvod je prostý: **gunzip** je pouze symbolickým linkem na `/bin/gzip`:

```
$ cd /usr/bin
 $ ls -l gunzip
 lrwxrwxrwx   1 root  root  9 Feb  2 09:45 gunzip -> /bin/gzip
```

Takže spuštění **gunzip**u je ve skutečnosti spuštěním **gzip**u pod jiným jménem. Program umí rozlišit jak byl vyvolán a zvolí správnou akci. V tomto případě bude **gzip** vědět, že byl volán jako **gunzip** a dekomprimuje soubor. Proto můžete používat pro dekompresi souborů **gzip** i **gunzip**.

Kapitoa 15. vi

Obsah

vi(1) je standardním Unixovým programem pro editaci textů. Jeho zvládnutí je pro systémové administrátory nezbytné. Existuje několik klonů či verzí programu **vi**: **vi** samotný, dále **elvis**, **vile**, a **vim**. Aspoň jeden z nich se vyskytuje snad v každé verzi Unixu, a tedy i Linuxu. Všechny jeho verze obsahují stejné základní vlastnosti a sadu příkazů, takže zvládnutí jedné verze by mělo usnadnit zvládnutí ostatních.

vi nabízí množství užitečných pomůcek jako je zvýrazňování syntaxe, formátování kódu, výkonné mechanismy pro vyhledávání a nahrazování, makra a další. Tyto vlastnosti jej činí obzvláště atraktivním pro programátory, webové vývojáře a podobně. Systémoví administrátoři ocení automatizaci a integraci s jakýmkoli disponibilním shellem.

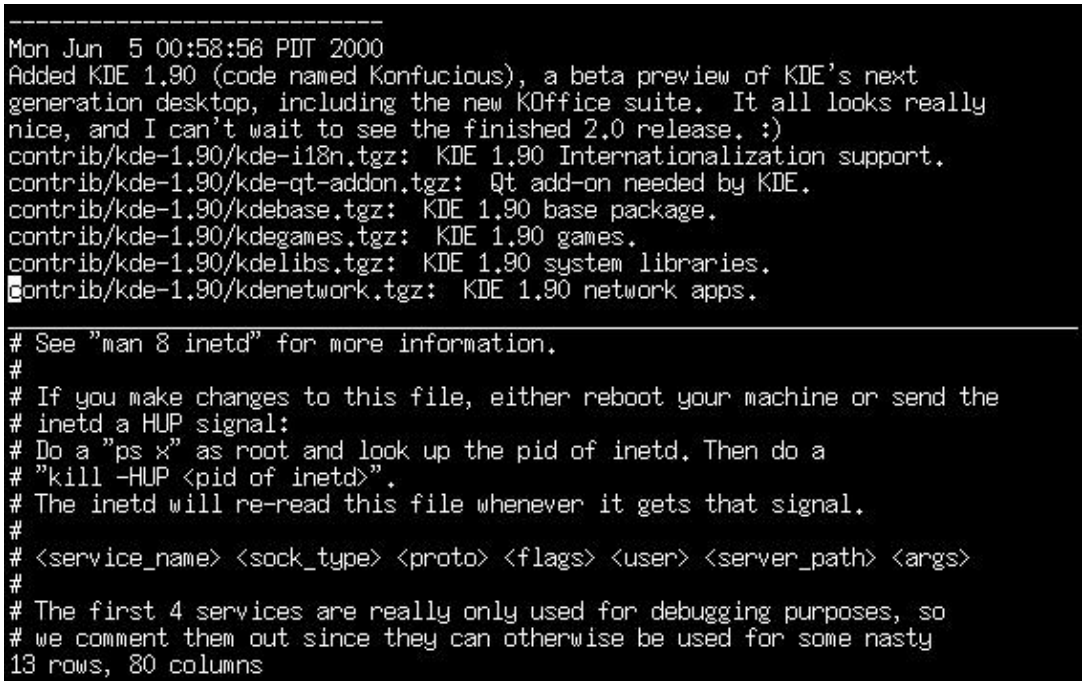
Výchozí verzí **vi** ve Slackware Linuxu je **elvis**. I další verze, včetně **vim** a **gvim** jsou k dispozici (pokud jste si je nainstalovali). **gvim** je X–window verzí **vim**u, která nabízí nástrojové lišty, odpojitelná menu a dialogové boxy.

Spouštění vi

vi může být spouštěn z příkazové řádky několika způsoby. Tím nejjednodušším je:

```
$ vi
```

Obrázek 15–1. Běh vi.



Takto nastartuje **vi** s prázdným bufferem. V tomto momentě uvidíte nejspíše prázdnou obrazovku. Je to "příkazový mód" (command mode) a čeká se na vás, že řeknete, co se má dělat. Módy editoru **vi** se probírají v sekci *Módy*. Abyste **vi** ukončili, napište:

```
:q
```

Pokud nebyly provedeny žádné změny v editovaném souboru, **vi** se bezprostředně ukončí. Pokud změny učiněny byly, bude vás **vi** varovat a současně vás poučí, jak toto varování přebít a problém ignorovat (napište :q!)

Také můžete spouštět **vi** s nějakým už existujícím souborem. Například soubor `/etc/resolv.conf` můžete otevřít takto:

```
$ vi /etc/resolv.conf
```

Dokonce můžete **vi** spustit tak, aby se zadaný soubor otevřel na nějakém konkrétním řádku. Například můžete spustit **vi** na 47. řádku souboru `/usr/src/linux/init/main.c`. Uděláte to takto:

```
vi +47 /usr/src/linux/init/main.c
```

vi zobrazí zadaný soubor a umístí kurzor na přísluš ný řádek. Pokud by číslo řádku bylo vyš š í, než jejich počet v souboru, umístí se kurzor na poslední řádek. Tato volba je obzvláš tě už itečná pro programátory, neboť jim umož ňuje skočit rovnou na řádek, kde se objevila chyba.

Kapitola 16. Správa Slackware balíčků

Obsah

Softwarový balíček je ranec spolu souvisejících programů (a jiných souborů), který je připravený k nainstalování. Když si stahujete archivy se zdrojovými kódy, musíte je konfigurovat, kompilovat a instalovat ručně. U softwarových balíčků to už bylo uděláno za vás. Jediné co musíte udělat, je balíček nainstalovat. Dalším milým rysem používání softwarových balíčků je snadnost jejich odstranění, nebo upgradu. Slackware je dodáván s programy splňujícími všechny potřeby správy balíčků. Balíčky můžete instalovat, odstraňovat, upgradovat, vytvářet a testovat velmi snadno.

Náhled do formátu balíčků

Než se budeme učit o utilitách, měli byste se dobře obeznámit s formátem Slackware balíčků. Balíček je v podstatě tar archiv, který byl ještě zkomprimován programem **gzip**. Balíček je sestaven tak, aby mohl být extrahován do root filesystému.

Zde je příklad fiktivního programu a jeho balíčku:

```
./
usr/
usr/bin/
usr/bin/makehejaz
usr/doc/
usr/doc/makehejaz-1.0/
usr/doc/makehejaz-1.0/COPYING
usr/doc/makehejaz-1.0/README
usr/man/
usr/man/man1
usr/man/man1/makehejaz.1.gz
install/
install/doinst.sh
```

Systém správy balíčků provede instalaci tak, že soubory z tohoto archivu extrahuje do kořenového adresáře. Dále systém udělá záznam do databáze balíčků. Záznam obsahuje informace o obsahu balíčku, aby mohl být později upgradován, nebo odstraněn.

Všimněte si podadresáře `install/`. Je to speciální adresář, který obsahuje post–instalační skript, který se jmenuje `doinst.sh`. Pokud balíčkový systém najde tento skriptový soubor, pak jej po nainstalování balíčku vykoná.

Do balíčku mohou být přidány i další skripty, ale o nich pojednáme podrobně až v kapitole o programu *makepkg*.

Kapitola 17. ZipSlack a BigSlack

Obsah

Co je to ZipSlack/BigSlack?

ZipSlack je speciální verze Slackware Linuxu. Je to předinstalovaná kopie Slackwaru, která může být spuštěna z vašeho DOS nebo Windows oddílu. Je to základní instalace, neobsahuje všechno, co kompletní Slackware. Jestliže od ZipSlacku očekáváte všechno, pak byste měli zkusit BigSlack.

ZipSlack se jmenuje podle formy ve které je distribuován – velký .ZIP soubor. Uživatelé DOSu and Windows pravděpodobně tento typ souborů znají. Jsou to kompresované archivy. ZipSlack archive obsahuje vše, co potřebujete k nastartování a běhu Slackwaru.

Je důležité zmínit, že ZipSlack a BigSlack jsou dost rozdílné od normální instalace. Přestože fungují stejně a obsahují stejné programy same and contain the same programs, their intended audiences and functions differ. Výhody a nevýhody ZipSlacku a BigSlacku jsou popsány níže.

Ještě jedna poznámka: Měli byste si vždy přečíst dokumentaci, která je v aktuálním ZipSlack nebo BigSlack adresáři. Obsahuje poslední informace, které se týkají instalace, bootování a obecně používání tohoto produktu.

Výhody

- Není nutné nové dělení vašeho disku na oddíly.
- Výborná cesta, jak se naučit Slackware Linux bez nutnosti klopýtat procesem instalace.

Nevýhody

- Používá DOS souborový systém, který je pomalejší než nativní Linux souborový systém.
- Nefunguje s Windows NT.

Předmluva

Operační systém Slackware Linux je výkonnou platformou pro počítače založené na procesorech Intel. Je navržen jako stabilní, bezpečný a funkční high-end server i výkonná pracovní stanice.

Záměrem této knihy je poskytnout vám úvod do operačního systému Slackware Linux. Není záměrem pokrýt každičký rys distribuce, ale spíše ukázat, čeho je schopný a dát vám základní vědomosti o práci tohoto systému.

Doufáme, že až získáte zkušenosti se Slackware Linuxem, stane se vám tato kniha užitečnou referenční příručkou. Taky doufáme, že ji budete půjčovat všem vašim přátelům, když se budou ptát na "ten cool operační systém Slackware Linux, ve kterym jedete."

I když tato kniha nemá na váš oblíbený román, snažili jsme se ji napsat co nejzáživněji. S trochou štěstí ji zfilmujeme. Především doufáme, že se z ní budete moci učit a shledáte ji užitečnou.

A nyní: Začněme show!

Konvence používané v této knize

Tato kniha je napsána v SGML s využitím DocBook 4.0 DTD. Použili jsme vestavěné prvky DocBooku pro jména souborů, příkazy i obsahy souborů. To nabízí konzistentní vzhled písem pro celou knihu. Budete potřebovat seznámit se s několika našimi zvyklostmi, dřív než budete pokračovat.

Kdykoli zmíníme příkaz, který máte spustit, bude to zapsáno takto:

```
command
```

Ve výjimečných případech může být příkaz delší, než se vejde na jednu řádku této knihy. Stane-li se to, rozdělíme příkaz do další řádky a použijeme backslash (obrácené lomítko) k indikaci, že příkaz pokračuje na další řádce. Zde je ukázka:

```
ifconfig eth0 192.168.1.10 broadcast 192.168.1.255 \
netmask 255.255.255.0
```

Jména souborů a adresářů jsou zmiňována po celé této knize. Vypadají takto:

```
filename
```

```
directory
```

Výpisy výstupu příkazů a obsahy konfiguračních souborů budou mít tento vzhled:

```
command output
```

Někdy, když vypisujeme příkazy, které máte spouštět, zobrazíme je jako by byly spouštěny z jednoduchého promptu. Je-li míněno, aby byl příkaz spouštěn obyčejným uživatelem, zobrazíme ho za promptem ve tvaru znaku dolar ($). Má-li být příkaz spouštěn rootem, zobrazíme jej za promptem ve tvaru znaku hash (#).

Glossary

Account
All of the information about a user, including username, password, finger information, UID and GID, and home directory. To create an account is to add and define a user.

Background
Any process that is running without accepting or controlling the input of a terminal is said to be running in the background.

Boot disk
A floppy disk containing an operating system (in our case, the Linux kernel) from which a computer can be started.

Compile
To convert source code to machine−readable binary code.

Daemon
A program designed to run in the background and, without user intervention, perform a specific task (usually providing a service).

Darkstar
The default hostname in Slackware; your computer will be called darkstar if you do not specify some other name.
One of Patrick Volkerding's development machines, named after Dark Star , a song by the Grateful Dead.

Desktop Environment
A graphical user interface (GUI) that runs atop the X Window System and provides such features as integrated applications, cohesive look−and−feel between programs and components, file and window management capabilities, etc. A step beyond the simple window manager.

Device driver
A chunk of code in the kernel that directly controls a piece of hardware.

Device node
A special type of file in the `/dev` filesystem that represents a hardware component to the operating system.

DNS
Domain Name Service. A system in which networked computers are given names which translate to numerical addresses.

Domain name
A computer's DNS name, excluding its host name.

Dot file
In Linux, files which are to be hidden have filenames beginning with a dot ('.').

Dotted quad
The format of IP addresses, so called because it consists of four numbers (range 0−255 decimal) separated by periods.

Dynamic loader
When programs are compiled under Linux, they usually use pieces of code (functions) from external libraries. When such programs are run, those libraries must be found and the required functions loaded into memory. This is the job of the dynamic loader.

Environment variable
A variable set in the user's shell which can be referenced by that user or programs run by that user within that shell. Environment variables are generally used to store preferences and default parameters.

Epoch
A period of history; in Unix, The Epoch begins at 00:00:00 UTC January 1, 1970. This is considered the dawn of time by Unix and Unix−like operating systems, and all other time is calculated relative to this date.

Filesystem
A representation of stored data in which files of data are kept organized in directories . The filesystem is the nearly universal form of representation for data stored to disks (both fixed and removable).

Foreground
A program that is accepting or controlling a terminal's input is said to be running in the foreground.

Framebuffer
A type of graphics device; in Linux, this most often refers to the software framebuffer, which provides a standard framebuffer interface to programs while keeping specific hardware drivers hidden from them. This layer of abstraction frees programs of the need to speak to various hardware drivers.

FTP
The File Transfer Protocol. FTP is a very popular method of transferring data between computers.

Gateway
A computer through which data on a network is transferred to another network.

GID
Group Identifier. The GID is a unique number attributed to a group of users.

Group
Users in Unix belong to groups , which can contain many other users and are used for more general access control than the existence of users alone can easily allow.

GUI
Graphical User Interface. A software interface that uses rendered graphical elements such as buttons, scrollbars, windows, etc. rather than solely text−based input and output

Home directory
A user's home directory is the directory the user is placed in immediately upon logging in. Users have full permissions and more or less free reign within their home directories.

HOWTO
A document describing how to do something, such as configure a firewall or manage users and groups. There is a large collection of these documents available from the Linux Documentation Project.

HTTP
The Hypertext Transfer Protocol. HTTP is the primary protocol on which the World Wide Web operates.

ICMP
Internet Control Message Protocol. A very basic networking protocol, used mostly for pings.

Kernel
The heart of an operating system. The kernel is the part that provides basic process control and interfaces with the computer's hardware.

Kernel module
A piece of kernel code, usually a driver of some sort, that can be loaded and unloaded from memory separately from the main body of the kernel. Modules are handy when upgrading drivers or testing kernel settings, because they can be loaded and unloaded without rebooting.

Library
A collection of functions which can be shared between programs.

LILO

The LInux LOader. LILO is the most widely−used Linux boot manager.

LOADLIN
LOADLIN is a program that runs under MS DOS or Windows and boots a Linux system. It is most commonly used on computers with multiple operating systems (including Linux and DOS/Windows, of course).

Man section
Pages in the standard Unix online manual ("man") are grouped into sections for easy reference. All C programming pages are in section 3, system administration pages in section 5, etc.

MBR
The Master Boot Record. A reserved space on a hard drive where information on what to do when booting is stored. LILO or other boot managers can be written here.

Motif
A popular programming toolkit used in many older X programs.

MOTD
Message of the Day. The motd (stored in Linux in `/etc/motd` is a text file that is displayed to all users upon logging in. Traditionally, it is used by the system administrator as a sort of bulletin board for communicating with users.

Mount point
An empty directory in a filesystem where another filesystem is to be mounted, or grafted on.

Nameserver
A DNS information server. Nameservers translate DNS names to numerical IP addresses.

Network interface
A virtual representation of a network device provided by the kernel. Network interfaces allow users and programs to talk to network devices.

NFS
The Network Filesystem. NFS allows the mounting of remote filesystems as if they were local to your computer and thus provides a transparent method of file sharing.

Octal
Base−8 number system, with digits 0−7.

Pager
An X program that allows the user to see and switch between multiple desktops.

Partition
A division of a hard drive. Filesystems exist on top of partitions.

PPP
Point−to−Point Protocol. PPP is used mainly for connecting via modem to an Internet Service Provider.

Process
A running program.

Root directory
Represented as /, the root directory exists at the top of the filesystem, with all other directories branching out beneath it in a file tree.

Root disk
The disk (usually fixed) on which the root directory is stored.

Routing table
The set of information the kernel uses in routing network data around. It contains such tidbits as where your default gateway is, which network interface is connected to which network, etc.

Runlevel
The overall system state as defined by init. Runlevel 6 is rebooting, runlevel 1 is single user mode, runlevel 4 is an X login, etc. There are 6 available runlevels on a Slackware system.

Secure shell
An encrypted (thus secure) method of logging in remotely to a computer. Many secure shell programs are available; both a client and server are needed.

Service
The sharing of information and/or data between programs and computers from a single server to multiple clients. HTTP, FTP, NFS, etc. are services.

Shadow password suite
The shadow password suite allows encrypted passwords to be hidden from users, while the rest of the information in the `/etc/passwd` file remains visible to all. This helps prevent brute−force attempts at cracking passwords.

Shell
Shells provide a commandline interface to the user. When you're looking at a text prompt, you're in a shell.

Shell builtin
A command built into the shell, as opposed to being provided by an external program. For instance, **bash** has a **cd** builtin.

Signal
Unix programs can communicate between each other using simple signals, which are enumerated and usually have specific meanings. **kill −l** will list the available signals.

SLIP
Serial Line Interface Protocol. SLIP is a similar protocol to PPP, in that it's used for connecting two machines via a serial interface.

Software package
A program and its associated files, archived and compressed into a single file along with any necessary scripts or information to aid in managing the installation, upgrade, and removal of those files.

Software series
A collection of related software packages in Slackware. All KDE packages are in the kde series, networking packages in the n series, etc.

Source code
The (more or less) human−readable code in which most programs are written. Source code is compiled into binary code.

Standard Error (stderr)
The Unix−standard output stream for errors. Programs write any error messages on stderr, so that they can be separated from normal output.

Standard Input (stdin)
The Unix−standard input stream. Data can be redirected or piped into a program's stdin from any source.

Standard Output (stdout)
The Unix−standard output stream. Normal text output from a program is written to stdout, which is separate from the error messages reported on stderr and can be piped or redirected into other programs' stdin or to a file.

Subnet
An IP address range that is part of a larger range. For instance, 192.168.1.0 is a subnet of 192.168.0.0 (where 0 is a mask meaning undefined); it is, in fact, the .1 subnet.

Superblock
In Linux, partitions are discussed in terms of blocks. A block is 512 bytes. The superblock is the first 512 bytes of a partition.

Supplemental disk

In Slackware, a floppy disk used during installation that contains neither the kernel (which is on the boot disk) nor the root filesystem (which is on the root disk), but additional needed files such as network modules or PCMCIA support.

Suspended process
A process which has been frozen until killed or resumed.

Swap space
Disk space used by the kernel as virtual RAM. It is slower than RAM, but because disk space is cheaper, swap is usually more plentiful. Swap space is useful to the kernel for holding lesser−used data and as a fallback when physical RAM is exhausted.

Symbolic link
A special file that simply points to the location of another file. Symbolic links are used to avoid data duplication when a file is needed in multiple locations.

Tagfile
A file used by the Slackware **setup** program during installation, which describes a set of packages to be installed.

Terminal
A human−computer interface consisting of at least a screen (or virtual screen) and some method of input (almost always at least a keyboard).

Toolkit, GUI
A GUI toolkit is a collection of libraries that provide a programmer with code to draw widgets such as scrollbars, checkboxes, etc. and construct a graphical interface. The GUI toolkit used by a program often defines its look and feel .

UID
User Identifier. A unique number that identifies a user to the system. UIDs are used by most programs instead of usernames because a number is easier to deal with; usernames are generally only used when the user has to see things happen.

VESA
Video Electronics Standards Association. The term VESA is often used to denote a standard specified by said Association. Nearly all modern video adapters are VESA−compliant.

Virtual terminal
The use of software to simulate multiple terminals while using only a single set of input/output devices (keyboard, monitor, mouse). Special keystrokes switch between virtual terminals at a single physical terminal.

Window manager
An X program whose purpose is to provide a graphical interface beyond the simple rectangle−drawing of the X Window System. Window managers generally provide titlebars, menus for running programs, etc.

Working directory
The directory in which a program considers itself to be while running.

Wrapper program
A program whose sole purpose is to run other programs, but change their behavior in some way by altering their environments or filtering their input.

X server
The program in the X Window System which interfaces with graphics hardware and handles the actual running of X programs.

X Window System
Network−oriented graphical interface system used on most Unix−like operating systems, including Linux.

Prev Summary | Home | Next The GNU General Public License

I. Úvod

Obsah

II. Instalace

Obsah

III. Konfigurace

Obsah

IV. Práce v Slackware Linuxu

Obsah

Co je to Slackware?

Slackware byla první linuxová distribuce, která dosáhla širokého využití. Byla nastartována Patrickem Volkerdingem na sklonku roku 1992. K Linuxu se Patrick dostal dík tomu, že pro jakýsi projekt potřeboval levný interpreter LISPu. V těch časech tu bylo poskrovnu distribucí. Patrik si vybral distribuci od Soft Landing Systems (SLS Linux).

Ovšem se SLS byly nějaké potíže, a tak Patrick začal opravovat drobné chyby, jakmile na ně přišel. Nakonec se rozhodl shromáždit všechny tyto úpravy do jeho vlastní distribuce, kterou měl pro sebe a své přátele. Tato soukromá distribuce získala rychle oblibu a Patrick ji dal k dispozici veřejnosti pod názvem Slackware.

Postupně Patrick přidal do této distribuce nové věci, jako například uživatelsky přívětivý instalační program založený na systému nabídek, a dále přidal koncept správy balíčků. To umožňuje uživatelům snadno přidávat, vyjímat, či upgradovat softwarové balíčky do/z jejich systému.

Open Source a Free Software

Uvnitř linuxové komunity pracují dvě hlavní ideologická hnutí. Hnutí Free Software, k němuž se dostaneme za chvíli, se snaží vytvářet veškerý software bez omezení daných intelektuálním vlastnictvím, které – jak věří – brání technickému zdokonalování a pracuje proti dobru komunity. Hnutí Open Source (otevřený zdroj) usiluje zhruba o totéž, ale přijímá k tomu mnohem pragmatičtější postoj, když preferuje v základu svých argumentů ekonomická a technická hlediska před morálními a etickými, jimiž se řídí hnutí Free Software.

Hnutí Free Software je vedeno nadací Free Software Foundation, což je nadační organizace pro projekt GNU. Free Software je více ideologií. Často používaným slovním obratem je free speech, not free beer (svobodná řeč, ale ne pivo zdarma – slovní hříčka která v češtině nefunguje). V podstatě je free software pokusem zajistit určitá práva jak vývojářům, tak i uživatelům. Tyto svobody zahrnují svobodu spouštět programy z jakýchkoliv pohnutek, svobodu studovat a upravovat zdrojový kód, svobodu redistribuovat zdroj a svobodu poskytovat (sdílet) všechny úpravy, které uděláte. K zajištění těchto svobod byla vytvořena GNU General Public Licence (GPL). GPL ve stručnosti stanovuje, že kdokoliv kdo distribuuje kompilovaný program, který je licencovaný pod GPL, musí rovněž poskytnout zdrojový kód. Také má volnost vytvářet modifikace tohoto programu, pokud jsou tyto modifikace rovněž zpřístupněny ve formě zdrojového kódu. To zabezpečuje, že jakmile je program jednou pro komunitu "otevřen", už nemůže být "uzavřen", s výjimkou souhlasu všech autorů každičkého kousku kódu (včetně úprav) tohoto programu. Většina linuxových programů je licencována pod GPL.

Je důležité poznamenat, že GPL neříká nic o ceně. Jakkoliv podivně to může znít, za free software můžete platit. "Volnost" spočívá ve svobodách, které které máte ke zdrojovému kódu, ne v ceně, kterou za software platíte. (I když, jakmile vám někdo prodal, nebo daroval kompilovaný program licencovaný pod GPL, je zavázán poskytnout rovněž jeho zdrojový kód.)

V čele hnutí Open Source, které je mladší, stojí Open Source Initiative – organizace existující výhradně proto, aby získávala podporu pro software s otevřeným zdrojem. To jest software, který má zdrojový kód dostupný stejně jako spustitelný program. Nenabízejí zvláštní licneci, ale místo toho podporují různé typy licencí dostupných pro otevřený zdroj.

Vedlejší ideou OSI je získat více organizací pro "open source" tím, že jim umožní napsat si vlastní licence pro otevření kódu a tyto licence jim certifikuje. Mnoho společností chce uvolnit zdrojový kód, ale nechtějí to dělat pod GPL. A protože nemůžou radikálně měnit GPL, je jim nabízena příležitost vytvořit si vlastní licenci a mít ji certifikovanou organizací OSI.

I když Free Software Foundation a Open Source Iniciative pracují tak, aby si pomáhaly, nejsou jedním a tím samým: Free Software Foundation používá specifickou licenci a poskytuje software pod touto licencí. Open Source Initiative usiluje o podporu pro všechny "open source" licence, včetně té od Free Software Foundation. Odlišné pohledy na způsob, jak dát zdrojový kód volně k dispozici občas tato dvě hnutí rozdělují. Ale ve skutečnosti tyto dvě ideologicky odlišné skupiny směřují k témuž cíli.

Online pomoc

Kromě dokumentace dodávané s operačním systémem Slackware Linux existuje ještě několik online zdrojů pomoci.

Website a fórum

www.slackware.com

Oficiální stránky Slackware Linuxu obsahují velké množství dokumentace. Jsou tu stránky pro počáteční pomoc, průvodce instalací, seznam často kladených otázek (FAQ) a spousty dalšího materiálu užitečného jak pro začátečníky, tak i pro zkušené uživatele.

Na website je rovněž k dispozici Slackware Fórum – sekce kde mohou uživatelé diskutovat o svých zkušenostech se Slackwarem a pomáhat si navzájem s různými problémy. Toto fórum se osvědčilo jako velmi populární a užitečný zdroj a mělo by být pravděpodobně vaší první zastávkou při hledání pomoci. Je tu totiž široké auditorium, které vaše otázka zasáhne, a vy pak máte velkou šanci, že váš problém bude rychle vyřešen. Pravděpodobně tu bude někdo, kdo měl tentýž problém, jako máte nyní vy. Prosíme, abyste ještě před zasláním vaší otázky do fóra toto fórum prohledali a pokusili se tam řešení vašeho problému najít. Je možné, že tam bylo totéž řešeno už dříve.

E–mailová podpora

Pokud si někdo koupí oficiální sadu CD, je oprávněn využívat bezplatnou instalační podporu prostřednictvím e–mailu. To praví, že my jsme ze staré školy. Uděláme co je v našich silách, abychom pomohli komukoliv, kdo nám e–mailne nějakou otázku. Prosíme, abyste ještě před mailováním zkoukli vaší dokumentaci a website (zvláště FAQ a Forum) – možná tak získáte ještě rychlejší odpověď. Navíc čím méně e–mailů musíme zodpovídat, tím rychlejší jsou naše odpovědi.

E–mailová adresa na technickou podporu je: <support@slackware.com>. Další e–mailové adresy a kontaktní informace jsou uvedeny na website.

Systémové požadavky a vlastní proces instalace

Poznámka: Protože nároky se s novými verzemi mění a vyvíjejí, nestyďte se podívat do "Slackware-HOWTO", které najdete na instalačním CD, nebo v kořenovém adresáři na ftp (byť je v angličtině).

Poznámka2: Během instalace (po nabootování) máte k dispozici několik virtuálních konzolí. Na té první probíhá samotný proces instalace. Na ostatních můžete libovolně pracovat. Instalujete-li si Linux poprvé, pak to asi nevyužijete, ale pokročilejší linuxman by tuto možnost neměl zapomínat. Například se můžete chtít během instalace kouknout přímo na instalační CD. Přijměte prosím informaci, že je namountováno v adresáři: /var/log/mount (kdo by ho tam hledal, že?:o). Cílový oddíl na kterém se instalace vytváří je připojený pod /mnt.

Jednoduchá instalace Slackwaru má tyto minimální nároky:

Tabulka 3-2. Systémové požadavky

Hardware	Požadavky
Procesor	386
RAM	16 MB
Diskový prostor	500MB
disketová jednotka	1.44 MB

Máte-li bootovatelné CD, nebudete nejspíš potřebovat disketovou jednotku. Pochopitelně pokud budete instalovat z CD, musíte mít CD mechaniku. Pro instalaci pomocí NFS (přes síť) budete potřebovat síťovou kartu. Na podrobnosti se podívejte do kapitoly *NFS*

Otázka diskového prostoru není tak jednoduchá. Na úplnou instalaci, budete potřebovat kolem 3 gigabajtů volného diskového prostoru (2GB pro Slackware a ještě trochu místa pro sebe). Většina uživatelů nedělá úplnou instalaci. Ve skutečnosti mnoho uživatelů provozuje Slackware na méně než 100MB.

Slackware může být nainstalován na systémy s menší RAM a menším hard diskem, ale tohle udělat vyžaduje trochu "kloubního mazu". Jste-li nachystáni na menší prácičku, koukněte do souboru `LOWMEM.TXT` (najdete ho v distribučním stromu). Je tam pro vás pár užitečných rad.

Softwarové skupiny

Kvůli zjednodušení byl Slackware historicky rozdělen do softwarových skupin. Kdysi se jim říkalo "diskové sady", protože byly určeny pro instalaci z disket. Dnes jsou softwarové skupiny používány hlavně k rozdělení softwarových balíčků do kategorií. V současnosti je disketová instalace dosud možná pro skupinu A a větší část skupiny N (viz níže).

Následuje stručný popis všech skupin softwaru.

Tabulka 3-3. Softwarové skupiny

Skupiny	Obsah
A	Základní systém. Obsahuje dostatek softwaru, aby se dal systém spouštět, mít textový editor a základní komunikační program.
AP	Rozličné aplikace, které nevyžadují X Window systém.
D	Programovací vývojářské nástroje. Kompilátory, debuggery, interpretery a man-stránky jsou všechny tady.
E	GNU emacs.
F	FAQ, HOWTO, a další dokumentace.
GNOME	Gnome desktop, knihovna GTK widgetů a GIMP.
K	Zdrojový kód kernelu Linuxu.
KDE	KDE desktop. X prostředí, které sdílí mnoho z vzhledu a dalších rysů s MacOS a Windows. Knihova Qt, kterou KDE potřebuje, je rovněž v této skupině obsažena.
N	Síťové programy. Démoni, mailovací programy, telnet, čtečky news, atd.
T	TeTeX - systém pro formátování dokumentů.
TCL	Jazyk Tool Command Language. Tk, TclX, and TkDesk.
X	Základ X Window Systému.
XAP	X aplikace, které nejsou součástí hlavních desktopových prostředí (např. Ghostscript a Netscape).
Y	Klasické textové orientované BSD hry

Způsoby instalace

Disketa

I když bývalo možné nainstalovat celý Slackware Linux z disket, rostoucí velikost softwarových balíčků tuto éru ukončila. Dnes už jen dvě skupiny softwaru mohou být takto instalovány: Skupina A je dosud plně instalovatelná z disket, a rověž větší část skupiny N. To vám dá naprosto základní systém, který můžete použít k instalaci zbytku distribuce protřednictvím sítě.

Poznamenejme, že diskety mohou být stále potřeba i při instalaci z CD-ROM, pokud tato jednotka není bootovatelná, a také při instalaci přes NFS.

CD-ROM

Máte-li bootovatelné CD, které je dostupné v oficiální sadě disků vydané Slackware Inc. (viz kapitola *Získání Slackwaru*), bude pro vás instalace z CD snazší. Pokud ne, budete muset bootovat z disket. Rovněž pokud máte speciální hardware, který činí problematickým použít kernel z bootovacího CD, budete potřebovat specializované diskety.

Je–li to třeba, koukněte do kapitol *Boot Disk* a *Supplemental Disk* na informace o tom, jaké diskety pro bootování vybrat a jak je vytvořit.

NFS

NFS (Network File System – síťový souborový systém) je způsob, jak zpřístupnit souborový systém vzdáleným počítačům. NFS instalace dovoluje instalovat Slackware z jiného počítače ve vaší síti. Stroj, z něhož se bude instalovat musí mít nakonfigurováno exportování distribučního stromu Slackwaru na stroj, na nějž se bude instalovat. To samozřejmě předpokládá jisté znalosti o NFS, které jsou popsány v podkapitole *NFS (Network File System)* v kapitole 5.

Je možné provést NFS instalaci i pomocí takových metod, jako je PLIP (přes paralelní port), SLIP a PPP (ne přes modemové připojení). Přesto doporučujeme použít síťovou kartu. Ona totiž instalace operčního systému přes port vaší tiskárny je velmi velmi pomalý proces.

Boot Disk (bootovací disketa)

Bootovací disketa je ta disketa, z níž nabootujete, abyste mohli začít s instalací. Obsahuje komprimovaný obraz (image) kernelu, který má za úkol ovládat hardware v průběhu instalace. Tuto disketu nebudete potřebovat, pokud budete bootovat z CD–ROM. Viz podkapitola *CD–ROM*. Obrazy bootovacích disket jsou umístěny v adresáři `bootdisks`, v distribučním stromu.

V tomto adresáři napočítáte hodně obrazů instalačních disket (cca 20). Jejich kompletní sezam i s podrobným popisem najdete v distribučním stromu v souboru `bootdisks/README.TXT`. Většině lidí bude vyhovovat obraz `bare.i` (pro IDE jednotky), nebo `scsi.s` (pro SCSI jednotky).

V podkapitole *Jak diskety vyrobit* najdete návod, jak si vyrobit diskety z jejich obrazů.

Po nabootování budete vyzváni ke vložení "root disku":

Root Disk

Root disketa obsahuje program setup a dále souborový systém, který je používán v průběhu instalace. Je rovněž vyžadována. Obrazy root disket jsou v adresáři `rootdisks` distribučního stromu.

V adresáři najdete především toto:

- `color.gz` – dříve používaný systém, který je tu k dispozici už je pro zatvrzelé nostalgiky.:o)
- `install.1 – install.5` – To jsou ty diskety, které budete potřebovat. Upozornění: Při instalaci je využíváno prostředí programu Dialog, který je choulositvý na mačkání kláves "mezi obrazovkami". Proto mačkejte s citem.:o)
- `rescue.dsk` – není pro instalaci, ale pro pozdější situace, kdy systém zkolabuje a potřebujete základní systém na disketě. Je tu jádro, všechny základní příkazy a programy včetně síťových utilit, editor vi. Zkrátka komplexní disketová mini–distribuce.

Supplemental (doplňkový) disk

Doplňková disketa je potřeba pro NFS instalaci, nebo když instalujete na systém se zařízeními PCMCIA. Obrazy doplňkových disket jsou ve stejném adresáři jako root–disky. Jejich jména jsou `network.dsk` a `pcmcia.dsk`.

Root disk vás bude o použití doplňkové diskety instruovat.

Jak diskety vyrobit

Když jste si vybrali správný obraz diskety, potřebujete jej překopírovat na disketu. Tento proces se liší podle toho, na jakém operačním systému se chystáte diskety vytvářet. Pokud jedete na Linuxu (nebo lépe řečeno na nějakém Unix–ovém OS), použijete příkaz **dd**(1). Například obraz diskety install.1 dostanete na disketu (předpokládejme, že ve vašem systému má označení `/dev/fd0`) takto:

```
# dd if=istall.1 of=/dev/fd0
```

Jedete–li na Microsoft OS, budete muset použít program **RAWRITE.EXE**. Najdete ho v distribučním stromu ve stejném adresáři, ve kterém jsou obrazy disket. Opět předpokládejme jméno vámi vybraného obrazu diskety `install.1`, a že disketová jednotka má označení A: Otevřte DOSové okno a napiš te:

```
c:\ rawrite a: install.1
```

Vytváření diskových oddílů

Upozornění: Zde se popisuje nejjednodušší varianta – instalace na prázdný disk. Pokud máte na disku například Windows a chcete je tam zachovat, musíte použít sofistikovanější návod – např. Installation–HOWTO, jehož český překlad najdete na http://qwert.cz/linux/.

Po nabootování z vámi zvoleného média budete potřebovat rozdělit váš harddisk na oddíly. Diskové oddíly jsou místem, kde bude vytvořen linuxový filesystém (souborový systém), a kam bude Slackware Linux nainstalován. Jako naprosté minimum doporučujeme vytvořit alespoň dva diskové oddíly: Jeden pro root filesystém (`/`) a jeden pro swapovací prostor.

Po načtení root disku se vám objeví přihlašovací výzva. Přihlaste se jako **`root`** (heslo se tu nezadává). Na výzvu (prompt) shellu spusťte buď **cfdisk**(8), nebo **fdisk**(8). Program **cfdisk** nabízí uživatelsky příjemnější prostředí než regulérní **fdisk**, ale postrádá některé funkce. Níže stručně popíšeme program **fdisk**.

Začněte spuštěním **fdisk**u na váš harddisk. V Linuxu nejsou harddisky označovány písmeny, ale jsou prezentovány jako soubory. První IDE harddisk (primary master) je `/dev/hda`, primary slave je `/dev/hdb`, a tak dále. SCSI disky se označují podobným způsobem takto: `/dev/sdX`. Takže spustíte si **fdisk** a zadáte mu jméno vašeho harddisku:

```
# fdisk /dev/hda
```

Stejně jako všechny dobré Unixové programy i **fdisk** s vámi komunikuje prostřednictvím promptu (čekali jste, že tam bude menu, že jo?). První věcí, kterou byste měli udělat je přezkoušení stávajících oddílů. Uděláme to zadáním **p** na prompt **fdisk**u:

```
Command (m for help): p
```

To vám zobrazí vš echny mož né informace o vaš ich současných oddílech. Větš ina lidí si pro instalaci vybere volný harddisk a potom vymaž e vš echny existující oddíly, které na něm byly, aby si tak udělali místo pro linuxové oddíly.

> JE VELMI DŮLEŽITÉ, ABYSTE SI UDĚLALI ZÁLOHU VŠECH INFORMACÍ Z DISKU, KTERÉ BUDETE CHTÍT UCHOVAT JEŠTĚ PŘED TÍM, NEŽ ZAČNETE S VYMAZÁVÁNÍM ODDÍLŮ.

Neexistuje snadný způsob, jak obnovit data z vymazaného oddílu. Takž e vž dycky zálohujte ješ tě před tím, než si s tím začnete hrát.

Když se podíváte na informace v tabulce oddílů, měli byste vidět číslo oddílu, velikost oddílu a jeho typ. Jsou tu i dalš í informace, ale těmi se teď nezabývejte. Chystáme se vymazat vš echny oddíly na tomto disku, abychom mohli vytvořit nové pro Linux. Zadáme příkaz **d** k jejich vymazání:

```
Command (m for help): d
Partition number (1-4): 1
```

Tento proces zopakujte pro vš echny oddíly. Po jejich vymazání se můž eme pustit do vytváření oddílů pro Linux. Rozhodli jsme se vytvořit jeden oddíl pro root filesystém a jeden pro swap. Nemá cenu se zabývat tím, ž e schéma rozdělení disku v Unixu je předmětem mnoha flame–wars, a ž e větš ina už ivatelů vám řekne jaký je ten *nejlepš í* způsob, jak to udělat. My radíme udělat pro začátek dva oddíly. Jeden pro root filesystém a jeden pro swapovací prostor. Po čase si sami vytvoříte rozdělovací schéma, které se pro váš systém bude nejlépe hodit.

Nyní vytvoříme ony oddíly pomocí příkazu **n**:

```
Command (m for help):n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (0-1060, default 0): 0
Last cylinder or +size or +sizeM or +sizeK (0-1060, default 1060): +64M
```

Měli byste se ujistit, ž e vytváříte primární oddíly. První oddíl bude slouž it pro swapovací prostor. Říkáme programu **fdisk**, aby vytvořil oddíl číslo 1 jako primární oddíl. Začínat bude na cylindru 0 a dále zadáme jeho velikost tím ž e napíš eme **+64M**. To nám dá 64 MB pro swap. (Jakou velikost swapovacího oddílu budete potřebovat doopravdy, závisí na velikosti vaš í operační paměti RAM. Obvykle se za rozumnou považ uje velikost swap oddílu o dvojnásobku velikosti RAM.) Pak zadáme primární oddíl číslo 2, který bude začínat na prvním volném cylindru a končit bude až na posledním cylindru disku.

```
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 2
First cylinder (124-1060, default 124): 124
Last cylinder or +size or +sizeM or +sizeK (124-1060, default 1060): 1060
```

A jsme téměř u konce. Ješ tě potřebujeme změnit typ prvního oddílu na typ č. 82 (Linux swap). Uděláme to příkazem **t**. Zadejte jej, pak vyberte první oddíl a napiš te **82**. Ješ tě před zapsáním provedených změn na disk byste se měli přesvědčit jak vypadá nová tabulka oddílů. Použ jte příkaz **p** k vypsání této tabulky. Pokud vš e vypadá v pořádku, zadejte příkaz **w**, který tabulku oddílů zapíš e na disk. Pak ukončete program **fdisk**.

Program setup

Jakmile jste si vytvořili oddíly, jste připraveni instalovat Slackware. Dalš ím krokem v instalačním procesu je spuš tění programu **setup**(8). To uděláte jednoduš e tak, ž e na prompt shellu napíš ete **setup**. Program **setup** se ovládá pomocí menu a použ ívá se k nainstalování softwarových balíčků Slackwaru a k různým nastavením vaš eho systému.

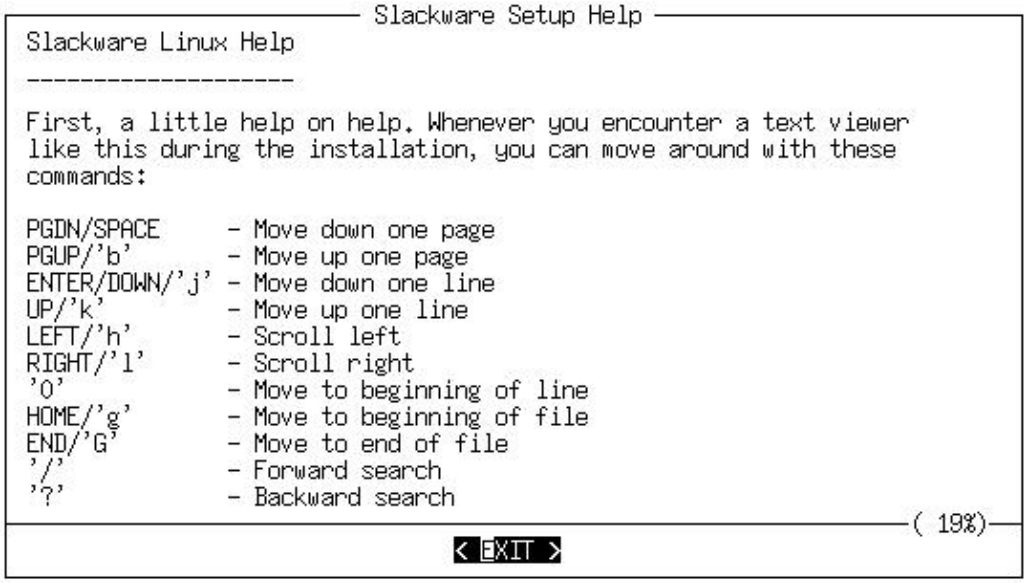
Dále pokračujeme nějak takhle: Projdete kaž dou polož ku menu programu **setup** v pořadí v jakém jsou uvedeny. (Zajisté máte volnost dělat to téměř v jakémkoliv pořadí, ale je tu mož nost, ž e to nebude moc dobře fungovat.) Polož ky menu vybíráme pomocí kláves se š ipkama nahoru a dolů; tlačítka

OK a Cancel mohou být vybrána pomocí šipek vlevo a vpravo. Náhradní možností je využít toho, že každá položka menu má svoji odpovídající písmenkovou klávesu, která je zvýrazněna v názvu položky. Přepínatelné volby (to jsou ty co jsou indikovány [X]) se přepínají pomocí mezerníku.

Tohle vše je rovněž popsáno v sekci help .

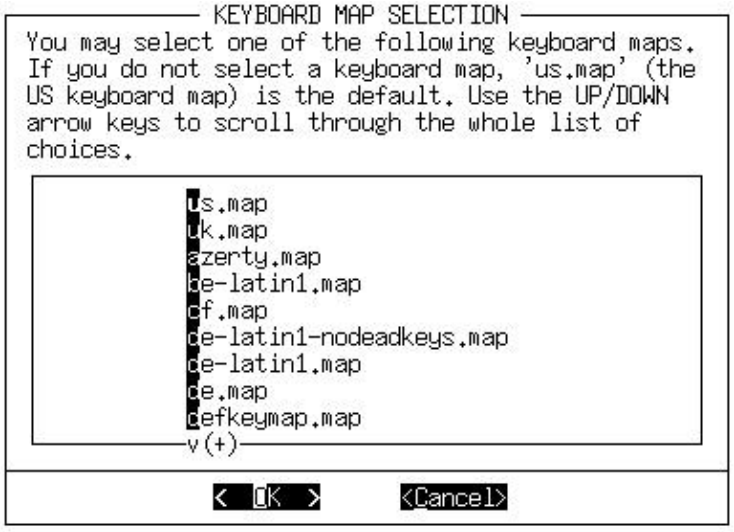
HELP

Pokud je toto vaše první instalace Slackwaru, asi byste se měli do této sekce podívat. Je tu popis každičké části programu **setup** (tak jak to právě popisujeme, ale méně podrobně) a instrukce pro navigaci celou instalací.



KEYMAP – mapa rozložení klávesnice

Pokud požadujete jiné rozložení kláves, než jaké nabízí US qwerty , měli byste tuto sekci navštívit. Nabízí množství alternativních rozložení kláves pro vaši klávesnici. Pro čechové: cz–us–qwertz, nebo méně písařská cz–lat2, nebo samozřejmě jiná, dle vaší libovůle. Obě zmíněně umožňují přepínání mezi českým a americkým rozložením kláves pomocí klávesy Pause/Break. Klávesnice qwertz má jako výchozí nastvení české, cz–lat2 americké.
Po výběru klávesnice se vám zobrazí okénko, kde si můžete klávesnici vyzkoušet. Nelekněte se, že vám tu nefungují česká písmena – je to proto, že ještě nemáte nainstalovaný český font pro obrazovku.:–p K tomu se dostanete až v závěru instalace. Během ní stejně české fonty potřebovat nebudete.
V budoucnu můžete měnit mapu klávesnice tak, že ji změníte v /etc/rc.d/keymap. Mapy klávesnic jsou uloženy v /usr/share/kbd/keymaps/i386/



ADDSWAP – zprovoznění swapovacího oddílu

Pokud jste zřídili swapovací oddíl (viz výše sekce *Vytváření diskových oddílů*), tato sekce vám jej umožní zprovoznit. Provede autodetekci a vypíše swapovací oddíly, které na vašem disku našel. Pak zadáte, který se má zformátovat a zprovoznit.

TARGET – zadání cílových oddílů

V této části instalace se mimo jiné zadávají diskové oddíly, na které se bude Slackware instalovat. K tomu je nutné přiřadit (namountovat) oddíly na disku filesystémům. Pokud jste postupovali dle výše uvedených doporučení a máte na disku dva oddíly, pak ten první, swapový, jsme vyřešili už v

předchozí části. Zbývá ten druhý, označovaný jako root (/).
V případě složitějšího členění disku asi víte co děláte a není nutné to tu rozebírat.
Vypíše se vám seznam oddílů na všem harddisku. Pro každý oddíl vám bude dána možnost jej zformátovat (a pokud ano, tak ještě zda provádět kontrolu na vadné bloky) a dále výběr velikostí inodů. Pro normální použití je defaultní velikost inode dobrá. Vyberete-li možnost provést kontrolu na vadné bloky, stihnete za tu dobu možná i celý oběd. Obvykle postačí pouhé zformátování (vytvoření souborového systému), které trvá jen pár vteřin.

První volbou v sekci target je výběr oddílu, kam se bude instalovat root (/) filesystém. Poté budete moci namapovat i další oddíly na vámi zvolené filesystémy. (Například budete-li chtít aby třetí oddíl, řekněme /dev/hda3, byl vaším home filesystémem. Toto je jen příklad; namapujte si oddíly jak se to hodí vám.)

SOURCE – zdrojová média

V sekci source (zdroj) zadáváte, z jakých médií budete Slackware instalovat. V současnosti si zde můžete vybrat ze čtyř různých zdrojů: Diskety, CD-ROM, NFS a před-mountovaný adresář.

Výběr disket zahájí vyžadování mnoha a mnoha disket. Tato volba vyžaduje velké množství času a trpělivosti, avšak je možná. Zapamatujte si, že musíte vyrobit diskety ještě před tím, než spustíte program setup.

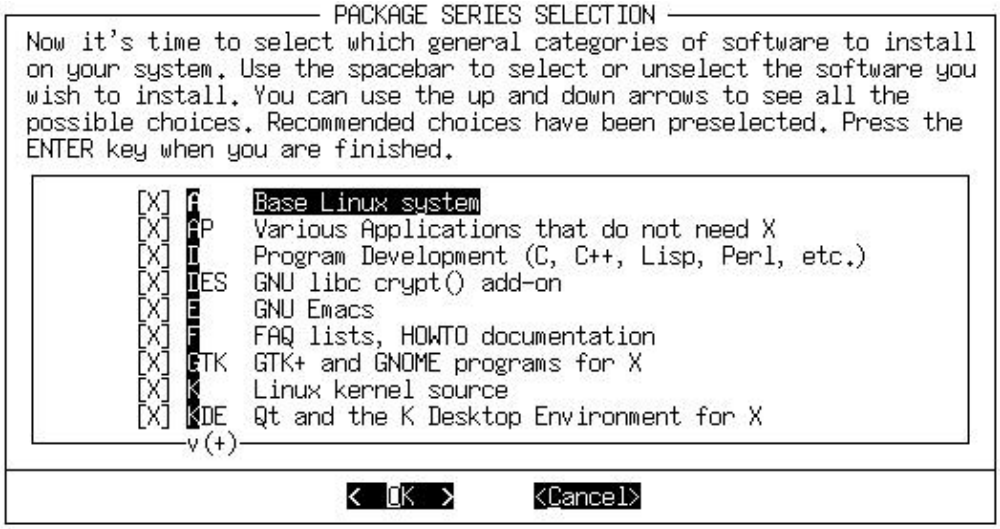
Výběr možnosti CD-ROM umožní instalaci z CD-ROM. Nabídne možnost vyhledání (scan) CD-ROM jednotky, nebo zobrazení seznamu, z něhož si můžete vaše zařízení vybrat. Ujistěte se, že máte Slackware CD vložené v mechanice ještě dřív než spustíte hledání. Poté, co program najde vaši CD-ROM jednotku, bude se vás ptát jestli chcete udělat slakware , nebo slaktest instalaci. Defaultní je slakware, která představuje standardní instalaci. Volba slaktest nainstaluje minimální množství softwaru na harddisk a větší inu ho pak užívá z CD. Aby tato možnost fungovala, budete potřebovat live CD, které je součástí oficiální CD sady. (poznámka: Takže při instalaci z jiného zdroje než oficiální sady se tato možnost ani nenabízí).

Volba NFS si vyžádá informace o síti a informace o vašem NFS serveru. NFS server musí být nastaven předem. Také si uvědomnte, že nemůžete používat hostnames, ale IP adresy, a to jak pro váš počítač, tak pro NFS server (na setup disku není žádná jmenná služba – name resolver).

Před-mountovaný adresář nabízí nejvíce flexibility. Tuto možnost můžete použít k instalaci z takových zařízení, jako jsou Jaz disky, NFS namoutované přes PLIP a nebo z FAT filesystému. Namountujte filesystém ještě před spuštěním setupu. V setupu pak zadáte, kde je filesystém namountován.

SELECT – výběr skupin softwaru k instalaci

Volba select vám umožňuje vybrat softwarové skupiny, které budete chtít instalovat. Tyto série jsou popsány v sekci *Softwarové skupiny*. Upozorňujeme, že sadu A nainstalovat musíte, abyste měli funkční základní systém. Všechny ostatní skupiny jsou volitelné.



INSTALL

Sem byste se měli dostat až poté, co jste proš li volbami target , source a select . Tady už budete vybírat jednotlivé softwarové balíčky z dříve zvolených softwarových skupin. Pokud jste to neudělali, budete vyzváni k tomu, abyste se vrátili a zkompletovali přípravu v předchozích částech setupu.
Tato volba vám dá vybrat ze š esti různých instalačních metod. *full*, *newbie*, *menu*, *expert*, *custom* a *tag path*.

Volba *full* vám naistaluje vš echny balíčky ze softwarových skupin, které jste vybrali v sekci select . Tady se už dál na nic neptá. Je to nejsnazš í instalační metoda, protož e nemusíte činit ž ádná rozhodnutí o tom, jaké balíčky se mají nainstalovat. Tato volba pochopitelně představuje největš í spotřebu prostoru na harddisku.

Následující volba je *newbie* (nováček). Tato volba nainstaluje automaticky vš echny vyž adované ("povinné") balíčky z vybraných skupin. U vš ech dalš ích balíčků se pak ptá, jestli jej chcte nainstalovat s mož ností odpovědět: Yes , No , nebo Skip . Yes a No dělají to, co obvykle. Skip přeskočí na následující softwarovou skupinu. Při této instalační metodě dostáváte navíc informaci o velikosti balíčku a popis toho, co dělá – tak se můž ete snáze rozhodnout, jestli ho budete chtít nainstalovat. Tuto volbu doporučujeme novým už ivatelům, protož e dává největš í jistotu, ž e nainstalujete vš e, co budete potřebovat. Ale protož e je tak upovídaná, je taky nejpomalejš í metodou.

Vobla *Menu* je rychlejš í a rozš ířenějš í verzí volby newbie. Pro kaž dou skupinu je zobrazeno menu, v němž si můž ete vybrat z nepovinných balíčků. Povinné (required) balíčky v tomto menu zobrazeny nejsou (ty se nainstalují automaticky). Zde už se nevypisují podrobné informace o balíčcích.

Pro pokročilejš í už ivatele se nabízí instalační volba *expert*. Ta nabízí úplnou kontrolu nad tím, které balíčky se budou instalovat. Dokonce můž ete zruš it výběr absolutně nezbytných ("povinných") balíčků a získat tak nekonzistetní systém. Na druhou stranu vidíte přesně, co se co vaš eho systému dostane.

Rovněž volby *custom* a *tag path* jsou určeny pokročilejš ím už ivatelům. Tyto volby umož ňují instalovat podle parametrů v tag souborech, které jste si vytvořili v distribučním stromu. To velmi zrychlí práci při stejné instalaci na velké množ ství počítačů. Více informací o tag souborech najdete v sekci *Vytváření Tagů a Tag souborů (pro setup)* v 16. kapitole.

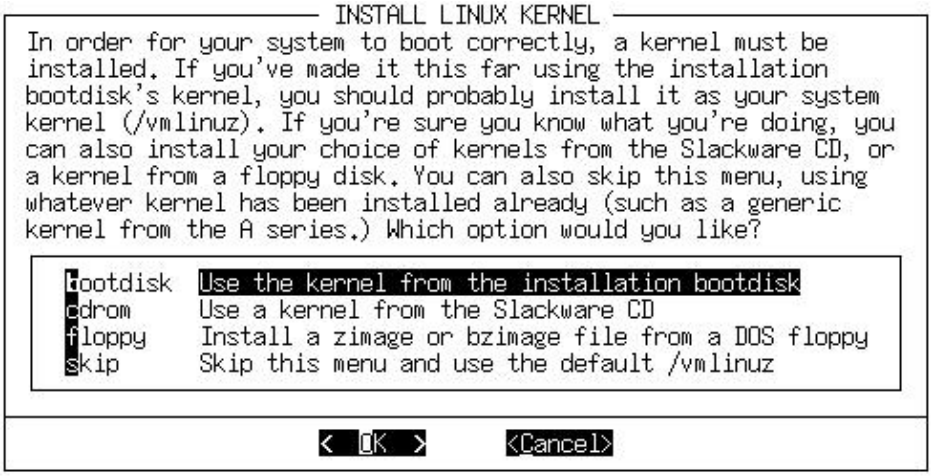
Po výběru instalační metody se stane jedna několika věcí. Pokud jste vybrali expert nebo menu, objeví se menu obrazovka, dovolující vám vybrat balíčky k nainstalování. Pokud jste vybrali full, začnou se balíčky hned instalovat na cílové oddíly. Vybrali–li jste newbie, budou se automaticky instalovat balíčky povinné, na volitelné budete dotazováni.

Také si uvědomte, ž e je mož né se při instalaci dostat mimo meze volného prostoru na disku. Abyste měli klid je lepš í vybrat při instalaci méně balíčků a dalš í pak případně doinstalovat do hotového systému. K tomu použ ijete bezproblémových nástrojů pro správu balíčků. Informace o nich jsou v Kapitole 16.

CONFIGURE – nastavení

Sekce configure umož ňuje provádět některá základní nastavení systému (poté, co byla dokončena instalace balíčků). Co tu uvidíte závisí i na tom, jaký software jste si nainstalovali. Vž dycky uvidíte následující:

Kernel selection (výběr jádra)
: Zde budete pož ádání, abyste vybrali kernel, který se má nainstalovat. Můž ete nainstalovat kernel z boot disku, který jste použ ívali při instalaci, z Slackware CD–ROMu, nebo z jiné diskety, kterou jste si (vž dy myslíte dopředu) předem připravili. Nebo můž ete vybrat "skip", v kterémž to případě bude nainstalován defaultní kernel.

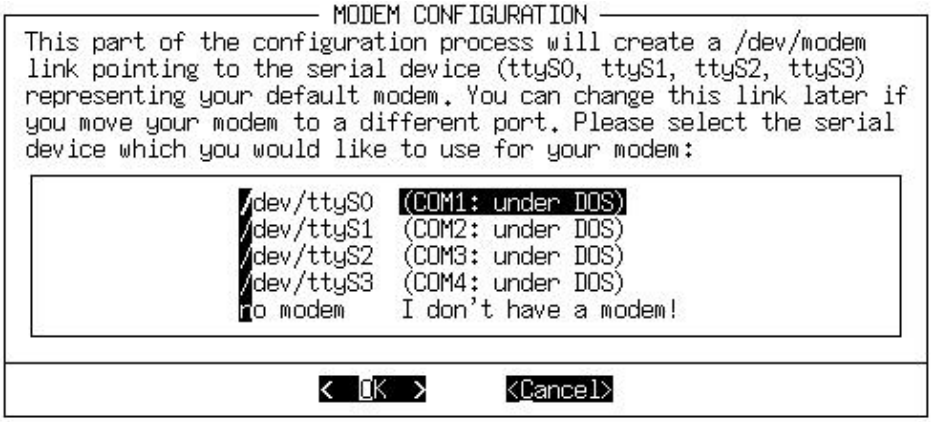


Make a boot disk (vytvoření bootovací diskety)

Vytvoření bootovací diskety pro budoucí využití je asi dobrý nápad. Dostanete na výběr formátování diskety a potom vytvoření jednoho ze dvou druhů boot disků. První typ, *simple*, jednoduše zapíše kernel na disketu. Mnohem flexibilnější (a velmi doporučovanou) volbou je *lilo*, která vytvoří bootovací disk s lilo zavaděčem. Více informací o "lilo" najdete v sekci *LILO* v 7. kapitole. Také můžete vybrat možnost **continue** (pokračovat) – pak se žádná bootovací disketa vytvářet nebude.

Modem

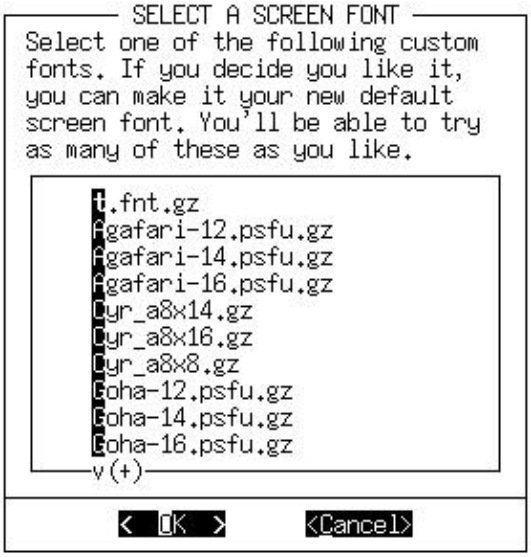
Zadáte, máte-li modem. A pokud ano, na kterém sériovém portu je připojen.



Následující konfigurační podsekce se mohou, ale nemusí objevit v závislosti na tom, jestli jste nainstalovali jim odpovídající softwarové balíčky. (**setup** je všestranně přizpůsobivý, jo.)

Font

Tato podsekce vám dává možnost vybrat si ze seznamu fontů určených pro konzoli. Čehůni si asi dají Lat2-16. Číslo na konci označuje velikost fontu. Můžete zkusit i menší, chcete-li. V zápětí dostanete možnost si font prohlédnout a v případě nespokojenosti si vyberete jiný. Pozor: Za chvíli budete nastavovat framebuffer, který vám umožní i v konzolovém režimu pracovat s rozlišením třeba 1024x768. Pokud si nyní vyberete drobný font, pak při jemnějším rozlišení obrazovky z něj budete mít font ještě prťavější. Je to sice legrační, ale nejde-li to přečíst, mrzí to.;-)
V budoucnu si můžete změnit obrazovkový font pomocí utility font-config, nebo ručně v /etc/rc.d/rc.font. Fonty hledejte v /usr/share/kbd/consolefonts/.



LILO

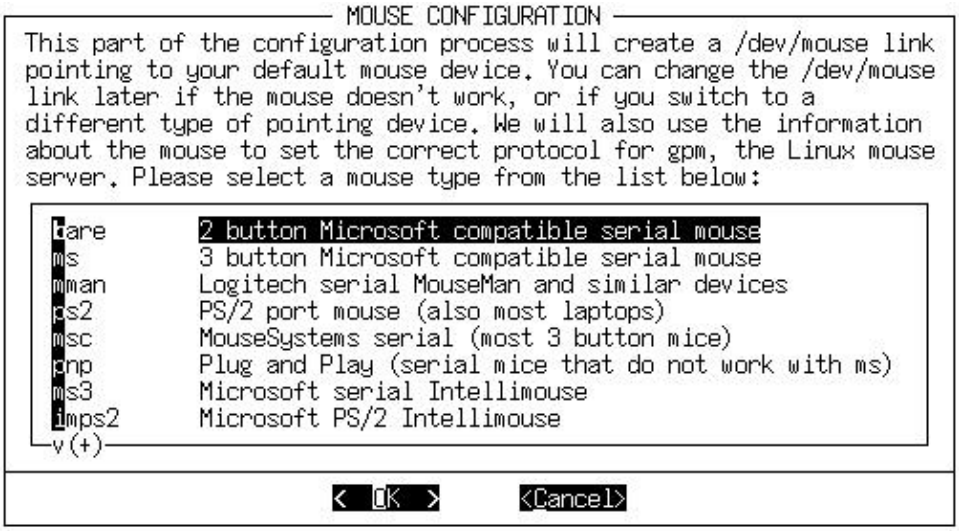
Tady se zadá zda a jak instalovat LILO (LInux LOader – linuxový zavaděč; viz sekce *LILO* v 7. kapitole). Pokud bude Slackware jediným operačním systémem ve vaš em počítači, pak vyberte volbu *simple*. Máte–li více operačních systémů (dual boot system), pak musíte vybrat volbu *expert*. Nahlédněte do sekce *Dual Booting* v 7. kapitole pro podrobnější informace. Třetí mož nost, *do not install* (neinstalovat) není doporučována, pokud nevíte přesně co děláte a víte proč nechcete nainstalovat LILO. Pokud jste zvolili volbu "expert", budete ješ tě dotázáni, kam se má LILO nahrát. LILO můž ete umístit do MBR (master boot record) vaš eho harddisku, nebo do superbloku vaš eho linuxového oddílu s root filesystémem, nebo na disketu.

Uvědomte si, prosím, ž e pokud v současnosti použ íváte bootovací zavaděč jiného operačního systému, je doporučeníhodné nainstalovat LILO do superbloku linuxového oddílu nebo na disketu. Instalace do MBR v tomto případě přepíš e bootovací zavaděč onoho jiného OS, což vám můž e dost zkomplikovat ž ivot.

Pokud jste vybrali volbu expert, projdeme si ji teď krok za krokem:
Tady budete poměrně pohodlnou metodou sestavovat konfigurační soubor pro zavaděč systémů z harddisku LILO – lilo.conf. V následujícím postupu budeme postupovat nejobvyklejš í cestou. V kaž dém případě se nedrž te následujícího postupu slepě, ale čtěte hláš ky na monitoru a řiďte se vlastním zdravým úsudkem.
Máte tu menu s mnoha volbami. Někde dole je i volba "view", která vám zobrazí aktuální stav lilo.conf. Dobré pro průběž nou kontorlu.
1. – volba begin. Vytvoří výchozí (prázdný) lilo.conf. Máte tu mož nost zadat volby pro append. Máte–li vypalovačku, pak zadáte hdc=ide–scsi (je–li vypalovačka na hdc). Dále se ptá, zda chcete použ ívat framebuffer (určite ano), a jaké rozliš ení pro něj použ ít (zvolte to, které na svém monitoru obvykle použ íváte). Později budete moci rozliš ení změnit ruční úpravou lilo.conf. Následuje dotaz, kam umístit lilo. Ve vš ech obvyklých případech vyberte MBR. Lilo umí zavádět i jiné OS (vč Windows). Posledním dotazem úvodní části je "čekání při bootu". Při startování systému se zobrazuje lilo prompt, který stanovenou dobu čeká na stisk klávesy levý shift. Stiknete–li jí, obdrž íte menu se vš emi operačními systémy, které je mož né nastartovat. Nestisknete–li jí ve stanoveném čase, zavede se defaultní systém. Volby v nabídce jsou celkem jasné, jen ta poslední – infinity – znamená "nekonečno"; tj. čeká se pořád.
2. – Linux – add a linux partition. Nyní přidáte do lilo.conf odkaz na diskový oddíl s linuxovým systémem. Dále jej musíte pojmenovat. Jméno nesmí obsahovat mezeru a musí být maximálně 8 znaků dlouhé. Jinak nenabootujete. Toto jméno se objeví ve výš e zmíněném startovacím menu.
3. – DOS – add a DOS partition. Tuto volbu budete potřebovat, máte–li na disku i Windows/DOS–ový oddíl. Postupujte stejně jako v předchozím bodě.
4. – Install Lilo. Předchozí postup ve větš ině případů dostačuje, takž e nyní už můž eme lilo nainstalovat. Před tím si ho mož ná ješ tě budete chtít prohlédnout volbou view.
Lilo je nainstalované.

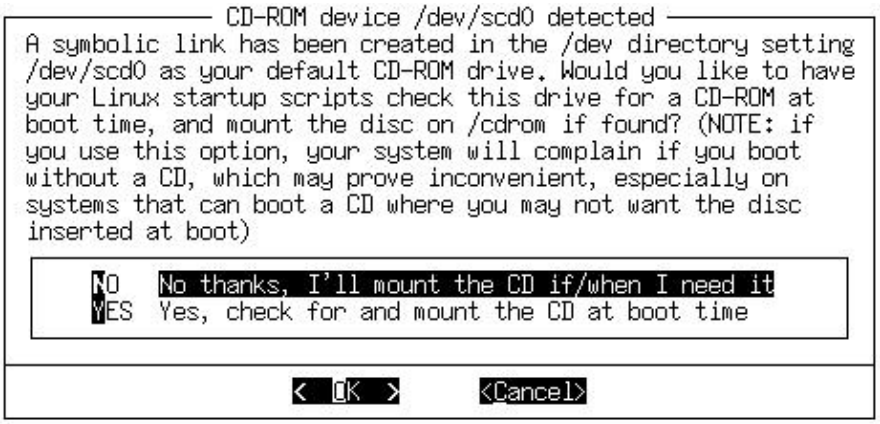
Mouse (myš)

Tato podsekce se jednoduš e ptá, jaký druh myš i máte. Vyberte si ze seznamu. Dalš í otázka se ptá, zda chcete při bootu zapínat **gpm**(8), což je už itčné rozš íření práce s myš í o mož nost copy–paste. Dejte Yes.



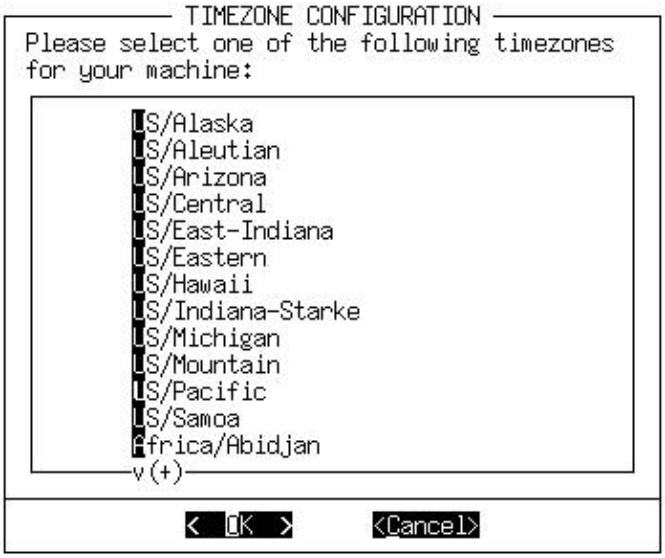
Network (síť)
Podsekci konfigurace sítě vlastně představuje program **netconfig**. Více informací o něm najdete v sekci *netconfig* v Kapitole 5

CD-ROM
Systém se ptá, pod jaký adresář se má připojovat CD mechanika. Obvykle to bývá /mnt/cdrom.

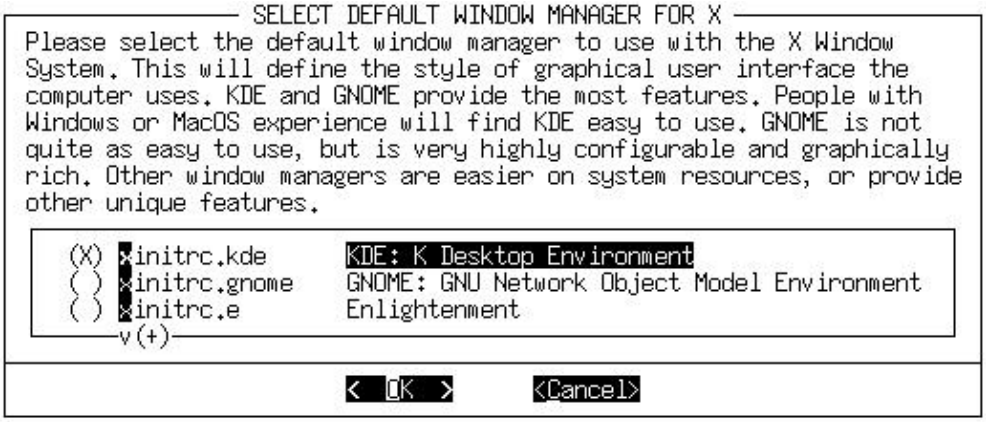


Hardware clock (hardwarové hodiny)
Tato subsekce se ptá, jestli hadrwarové hodiny ve vaš em počítači jsou nastaveny na koordinovaný světový čas (UTC či GMT). Větš ina PC není, takž e nejspíš řeknete ž e ne.

Timezone (časové pásmo)
Tato volba je vskutku přímočará: Jste pož ádáni abyste vybrali časové pásmo, v němž se nalézáte. Pokud operujete v Zulu čase, je nám velmi líto; (extrémně dlouhý) seznam je abecedně řazený a vy jste až na konci.



X Window Manager (okenní správce systému X)
Tady si můžete vybrat defaultní window manager pro X−ka. Více informací o X a okenních manažerech najdete v Kapitole 6



Nastavení hesla pro superuživatele root
Je dobré heslo nastavit. Při jeho psaní se vám nebude na monitoru nic vypisovat (ani hvězdičky). Systém pak heslo prověří a bude−li se mu zdát moc jednoduché, trochu vám vyčiní a dá vám možnost zadat jiné heslo. Budete−li vytrvalí a zadáte znovu své oblíbené jednoduché heslo, systém se vzdá a umožní vám to. Pak zadáte heslo ještě jednou, pro kontrolu.

EXIT

Samotná existence této sekce v této knize je urážkou vaší inteligence. Poníženě se omlouváme a prosíme o vaše odpuštění. (Nicméně, až budete rebootovat, vyndejte si CD z mechniky...;−)

Shrnutí

Nyní byste měli mít naistalovaný Slackware Linux ve vaš em systému. Navíc jste se více seznámili s rozdělováním disků na oddíly, se softwarovými balíčky, s programem **setup** a s některými jednoduchými konfiguračními volbami. S těmito znalostmi byste měli být připraveni na práci s dokončením konfigurace vaš eho systému.

Vybíráme si jádro

Jádro (kernel) je ta část operačního systému, která poskytuje přístup k hardwaru, správu procesů a celkovou kontrolu systému. Kernel obsahuje podporu pro vaše hardwarová zařízení, takže výběr toho správného pro váš systém je velmi důležitý krok.

Slackware nabízí okolo šedesáti předkompilovaných jader, ze kterých si můžete vybírat; každé se standardní sadou ovladačů a dodatečnými specifickými ovladači. Můžete provozovat jedno z těchto předkompilovaných jader, a nebo si můžete sestavit svoje vlastní jádro ze zdrojového kódu. V obou případech se musíte ubezpečit, že vaše jádro obsahuje hardwarovou podporu vyhovující potřebám vašeho systému.

Adresář /kernels na Slackware CD–ROM

Předkompilovaná jádra Slackwaru jsou dostupná v adresáři `/kernels` na Slackware CD–ROM nebo v FTP sajtu v hlavním adresáři Slackwaru. Dostupná jádra se průběžně mění, jak jsou vydávány nové verze, takže nejspolehlivějším zdrojem dokumentace je ta, kterou najdete rovněž v tomto adresáři. Adresář `/kernels` obsahuje pro každý dostupný kernel podadresář. Podadresáře mají stejná jména jako jejich doprovodné boot disky. V každém podadresáři najdete následující soubory:

Soubor	**Účel**
`System.map`	Systémová mapa pro toto jádro
`bzImage` (or `zImage`)	Vlastní obraz (image) jádra
`config`	Zdrojový konfigurační soubor pro toto jádro

Abyste mohli kernel používat, zkopírujte soubory `System.map` a `config` do adresáře `/boot`, a obraz jádra zkopírujte do adresáře `/vmlinuz`. Pak spusťte **/sbin/lilo**(8), aby se nainstalovalo LILO pro nové jádro, a nakonec rebootujte svůj systém. A to je pro instalaci nového jádra vše.

Jádra, jejichž název končí ".i" jsou IDE jádra. To znamená, že v základu tato jádra neobsahují žádnou podporu pro SCSI. Jádra s názvem zakončným ".s" jsou SCSI jádra. Ta obsahují kompletní podporu pro IDE – stejně jako ".i" – a navíc ještě podporu pro SCSI.

Kompilace jádra ze zdrojového kódu

Noví uživatelé si často kladou otázku, jestli by si měli zkompilovat vlastní jádro. Odpověď zní: "Určitě možná". Jsou případy, kdy budete kompilaci jádra specifického pro váš systém potřebovat. Většina uživatelů může používat předkompilované jádro se zaváděnými moduly jádra, aby dosáhli plně funkčního systému. Kompilaci jádra budete muset podstoupit v případě, že upgradujete verzi jádra na takovou, kterou Slackware dosud nenabízí, nebo pokud máte patchovaný zdroj jádra se speciální podporou zařízení, která není v nativním zdroji jádra obsažena.

Sestavení vašeho vlastního jádra není tak těžké. Prvním krokem je ujistit se, že máte zdroj jádra nainstalovaný ve vašem systému. Ujistěte se, že jste nainstalovali balíčky ze skupiny K. Také se budete muset ubezpečit, že máte nainstalovanou skupinu D; zvláště pak překladač jazyka C, GNU make a GNU binutils. Obecně je dobré mít nainstalovanou skupinu D kompletní, pokud plánujete provádění čehokoliv souvisejícího s vývojařinou. Teď jsme připraveni začít sestavovat jádro. Přepneme se do superuživatelského módu a přejdeme do adresáře se zdrojovými kódy jader:

```
$ su -
Password:
# cd /usr/src/linux
```

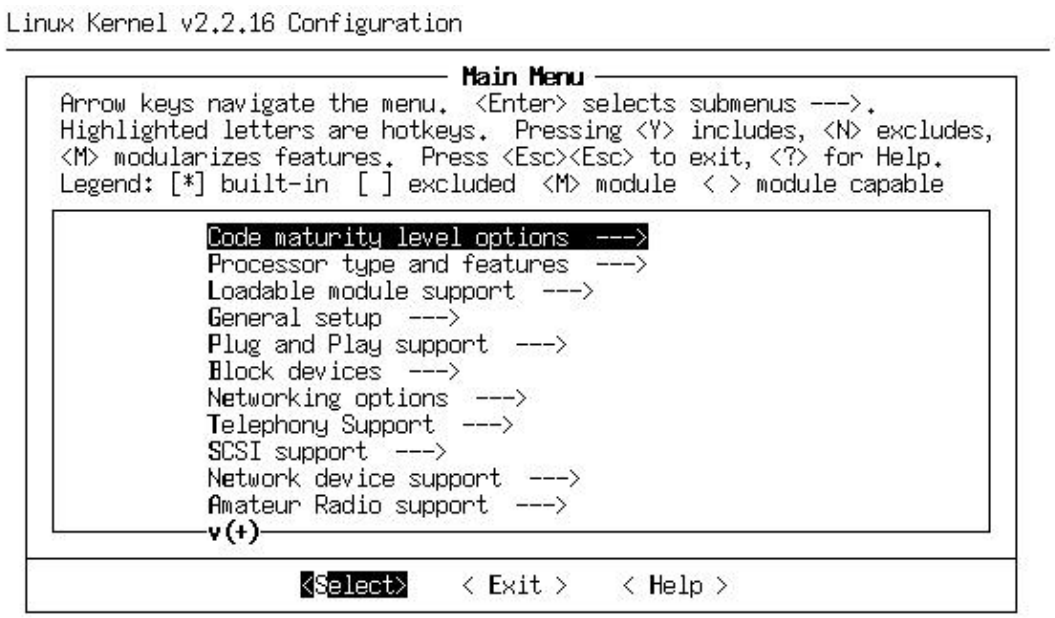
Prvním krokem je uvést zdroj jádra do výchozího stavu. To se dělá tímto příkazem:

```
# make mrproper
```

Teď můžete konfigurovat jádro pro váš systém. V současnosti máte k dispozici tři způsoby, jak to udělat. Prvním je původní, textově založený systém dotaz–odpověď (make config). Klade spoustu otázek a potom sestavuje kofigurační soubor. Problémem této metody je, že když něco zmastíte, musíte začít úplně znovu. Nejoblíbenější metodou většiny uživatelů je nabídkově orientovaná metoda (make menuconfig). A nakonec třetí možnost, rovněž nabídková, ale pro systém X–window (make xconfig). Vyberte si metodu, která vám vyhovuje nejvíc zadáním jednoho z příkazů:

```
# make config       (textové založená verze dotaz-odpověď)
# make menuconfig   (nabídkově orientovaná, textová verze)
# make xconfig      (Na X-založená verze, musíte být v X než ji spustíte!)
```

Obrázek 4–1. Nabídkami ovládaný program pro konfiguraci jádra.



Noví už ivatelé pravděpodobně shledají jako nejsnazš í variantu s **menuconfig**. Jsou tu nápovědné obrazovky, které nabízejí vysvětlující popis jednotlivých částí jádra. Po nastavení vaš eho jádra opusť te konfigurační program. Ten pak zapíš e nezbytné konfigurační soubory. Nyní můž eme připravit zdrojový strom na sestavování:

```
# make dep
# make clean
```

Dalš ím krokem je kompilace jádra. Nejdříve zkuste příkaz **zImage**. Je–li váš kernel hodně velký, pak tento program klekne. Nic se nebojte, pořád ješ tě můž ete sestavit jádro příkazem **bzImage**

```
# make zImage      (tohle zkuste jako první)
# make bzImage     (pokud make zImage zhavaruje, použijte tento příkaz)
```

Kompilace trvá dost dlouho – v závislosti na výkonu procesoru. V průběhu sestavování uvidíte zprávy, které vypisuje překladač. Po sestavení obrazu jádra budete ješ tě potřebovat sestavit ty části kernelu, které jste při konfiguraci označili jako moduly.

```
# make modules
```

A teď můž eme nainstalovat kernel a moduly, které jste právě zkompilovali. K nainstalování kernelu do systému Slackware musíte spustit následující příkazy:

```
# mv /vmlinuz /vmlinuz.old
# cat arch/i386/boot/zImage > /vmlinuz
# mv /boot/System.map /boot/System.map.old
# cp System.map /boot/System.map
```

Pokud jste sestavovali velké jádro, nahraďte výš e uvedený **zImage** za **bzImage**. Dále budete potřebovat editovat `/etc/lilo.conf` a přidat do něj sekci umož ňující nabootovat staré jádro pro případ, ž e by to nové nepracovalo dobře. Až to uděláte, spusť te příkaz **/sbin/lilo**, aby se nainstaloval nový bootovací blok. Teď můž ete rebootovat s vaš ím novým jádrem.

Použ ívání modulů jádra

Moduly jádra jsou jen jiným jménem pro ovladače zařízení, které mohou být přidávány do běž ícího kernelu. Umož ňují vám rozš ířit kernel o podporu hardwaru bez nutnosti vybírat jiné jádro, nebo si sestavovat vlastní.

Moduly mohou být zaváděny a uvolňovány kdykoliv, a to i při běž ícím systému. To systémovým administrátorům usnadňuje upgradování ovladačů. Nový modul můž e být přelož en, starý vyjmut a nový zaveden; to vš e bez rebootování stroje.

Moduly jsou ulož eny v adresáři `/lib/modules/<kernel version>`. Mohou být zaváděny v průběhu bootování prostřednictvím souboru `rc.modules`. Tento soubor je velmi dobře okomentářován a nabízí vysvětlivky pro hlavní hardwarové komponenty. Abyste viděli, které moduly jsou v tuto chvíli aktivní, použ jte příkaz **lsmod**(1):

```
# lsmod
Module                  Size  Used by
parport_pc              7220   0
parport                 7844   0  [parport_pc]
```

Jak vidíte, mám zavedený pouze modul pro paralelní port. K vyjmutí modulu použ ijte příkaz **rmmod**(1). Moduly mohou být zaváděny pomocí příkazů **modprobe**(1) nebo **insmod**(1). Příkaz **modprobe** je obvykle bezpečnějš í protož e zavede i vš echny moduly, na nichž je ten, který se pokouš íte zavést, závislý.

Mnoho uživatelů nikdy nemusí zavádět či vyjímat moduly ručně. Používají auto–zavaděč pro správu modulů jádra. Jediné co pro to musíte udělat, je odkomentářovat řádek **/sbin/kerneld**(8) v souboru `/etc/rc.d/rc.modules` a auto–zavaděč bude startovat. Převezme péči o zavádění a vyjímání modulů tak, jak se objevují požadavky na ně. Požadavkem rozumíme pokus o přístup na dané zařízení.

Více informací najdete v manuálových stránkách pro každý z těchto příkazů a v souboru `rc.modules`.

Předchozí Konfigurace systému | Výchozí Nahoru | Další Shrnutí

Shrnutí

Nyní byste měli dobře znát příkazy k prohledávání souborového systému, organizaci souborového systému a konfigurační soubory v adresáři `/etc`. Tyto dovednosti budou nesmírně užitečné, až se budete učit další věci o systému. A navíc byste nyní měli vědět, jak nakonfigurovat a zkompilovat jádro ze zdorjového kódu.

Síťové utility

ifconfig

Nyní, když váš kernel může hovořit s vaším síťovým hardwarem, co je potřeba je způsob pro software jak říct kernelu aby procházel nějaké informace a naopak. Potřebujeme nastavit rozhraní. Potřebujeme **ifconfig**(8).
*Now that your kernel can talk to your network hardware, what's needed is a way for software to tell the kernel to pass some information along, and vice versa. We need to configure an interface. We need **ifconfig**(8).*

Příkaz **ifconfig** se asi nejlépe naučí na příkladu. Snad byste se měli i kouknout do souboru `rc.inet1` (popsaný v sekci *rc.inet1*) a podívat se, jak je to tam uděláno. Nejjednoduší a nejobecnější případ vypadá nějak takto:

```
# ifconfig eth0 192.168.1.10 broadcast 192.168.1.255 \
netmask 255.255.255.0
```

Tento řádek spojuje eth0 (první ethernetové rozhraní; pro token ring používáme tr0, ppp používá ppp0, atd.) s IP adresou 192.168.1.10, s vysílací (broadcast) adresou 192.168.1.255 (celá podsíť 192.168.1) a se síťovou maskou 255.255.255.0 (indikující, že ve všechny první tři části "tečkované čtveřice" IP adresy odkazují na síť, zatímco poslední část, .10 odkazuje na hostitele). Jestliže neděláte něco "funky", můžete téměř vždy použít jako broadcast adresu první tři části vaší IP adresy následované 255. Rovněž téměř vždy můžete jako síťovou masku (netmask) použít 255.255.255.0. Pokud *děláte* něco "funky", pravděpodobně toho znáte dost a tato část knihy pro vás není příliš užitečná.

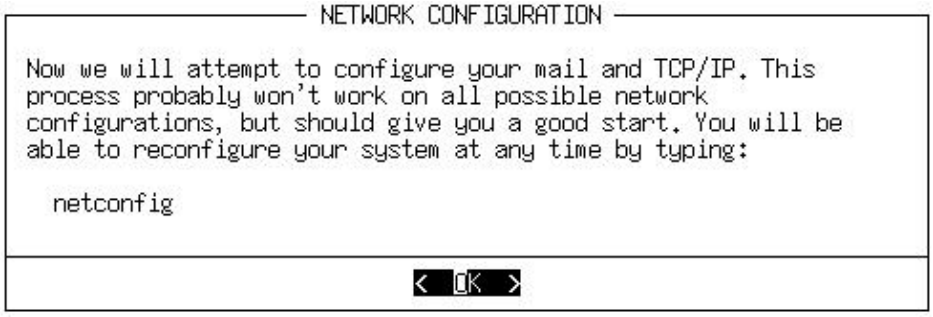
Příkaz **ifconfig** můžete také používat k tomu, abyste zjistili aktuální nastavení. Spusťte jej bez jakýchkoliv voleb či parametrů a získáte výpis všech vašich síťových rozhraní a jejich nastavení.

route

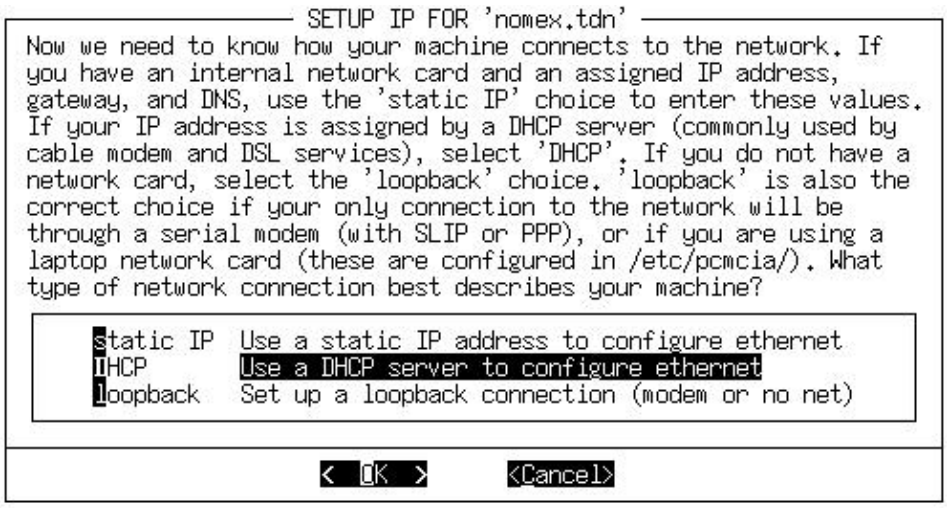
Aby kernel věděl kam posílat která data, udržuje si směrovací (routovací) tabulku. Nebudeme to tu popisovat nijak detailně. Směrovací tabulku si můžete prohlédnout spuštěním **/sbin/route**(8). Přidáte-li parametr **`route -n`**, dostanete směrovací tabulku s IP adresami namísto jmen. To se může hodit v situaci, kdy máte problémy s přístupem k jmennému serveru, nebo když vás nezajímá iluzorní svět doménových jmen. Naštěstí, máte-li jednoduchá síťová nastavení (a většina lidí má), jádra od verze 2.2 automaticky potřebnou routovací tabulku vytvoří za vás.

netconfig

Program **netconfig** je součástí programu "setup" užívaného ve Slackwaru. Nicméně stejně jako většina částí setupu i on může být spuštěn zcela samostatně. Program **netconfig** je velmi přímočarý a provede vás nastaveními pro základní připojení k síti. Je to velice dobrý program, zvláště když nic moc nerozumíte síťovým rc souborům. Spustíte-li **netconfig**, budete přivítáni touto obrazovkou:



Pak budete vyzváni k zadání hostname (jméno hostilete) a domain name (jméno domény). Můžete si tam zadat v podstatě co chcete, pokud ovšem nenastavujete server, nebo počítač, který bude využíván mnoha lidmi. Následně budete požádání abyste řekli, zda budete používat statickou IP adresu, DHCP, nebo jen loopback.



Pokud se nechystáte připojovat do sítě, vyberte loopback (pp: ten vyberte i v případě, že se budete připojovat jen do internetu přes modem). Jestliže nastavujete počítač, který bude připojen na univerzitní, nebo velkou firemní síť, budete nejspíš zadávat možnost DHCP. V ostatních případech vyberte statickou IP adresu.

Pokud jste nevybrali možnost "Statická IP adresa", jste pro tuto chvíli hotovi. Pokud jste vybrali statickou IP adresu, pak budete muset ještě zadat IP adresu vašeho počítače, masku sítě, broadcast adresu a adresu jmenného serveru. **netconfig** vám poradí, jak tato čísla vypočítat

Následuje dotaz, zda používat nameserver. Nevíte-li o co jde, dejte ne. Tip: Můžete si nainstalovat nameserver Bind a používat ho jako name-cash. To vám urychlí práci s internetem i při připojování přes modem.

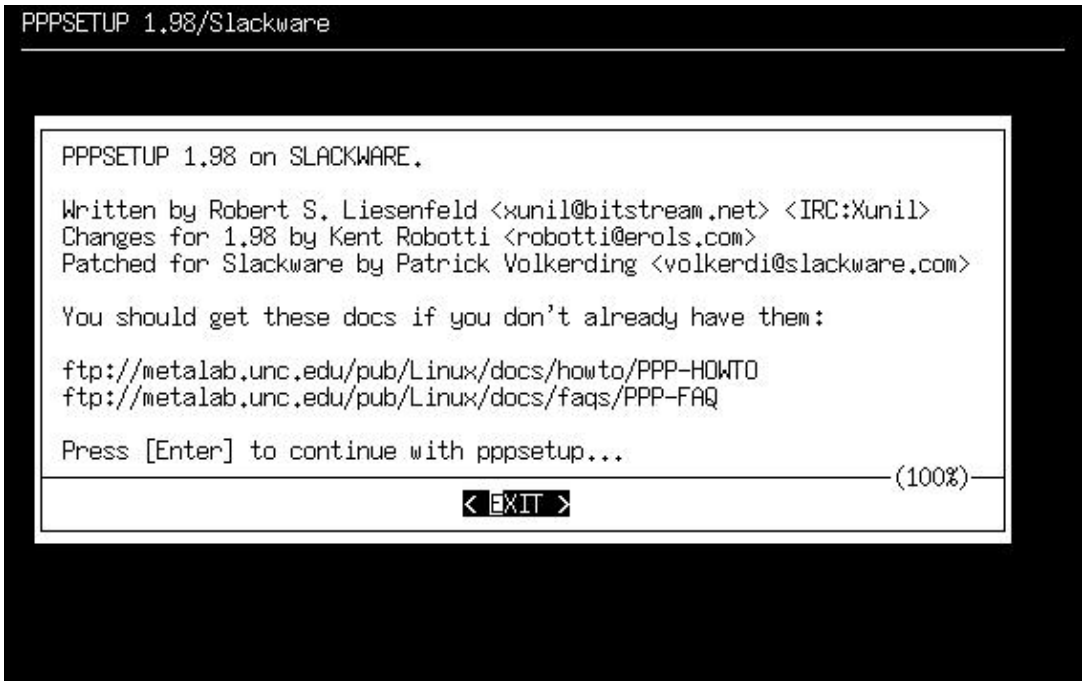
Další dotaz zjišťuje, zda se má netconfig pokusit identifikovat vaší síťovou kartu. Dáte-li probe, proběhne test. Pokud se kartu identifikovat podaří, vypíše se o tom hláška s uvedením jména ovladače pro tuto kartu.

Shrnutí – vypíše se souhrn všech nastavení, která jste dosud provedli. Pokud jej shledáváte v pořádku, odsouhlaste jej. V opačném případě dostanete možnost si celou konfiguraci zopakovat.

Sendmail. To je program, který se bude starat o odesílání vaší pošty (je to smtp server). Vy tu jen zadáváte, co od něj budete vyžadovat. Téměř určitě to bude první volba z nabídky – pouhé "smtp".

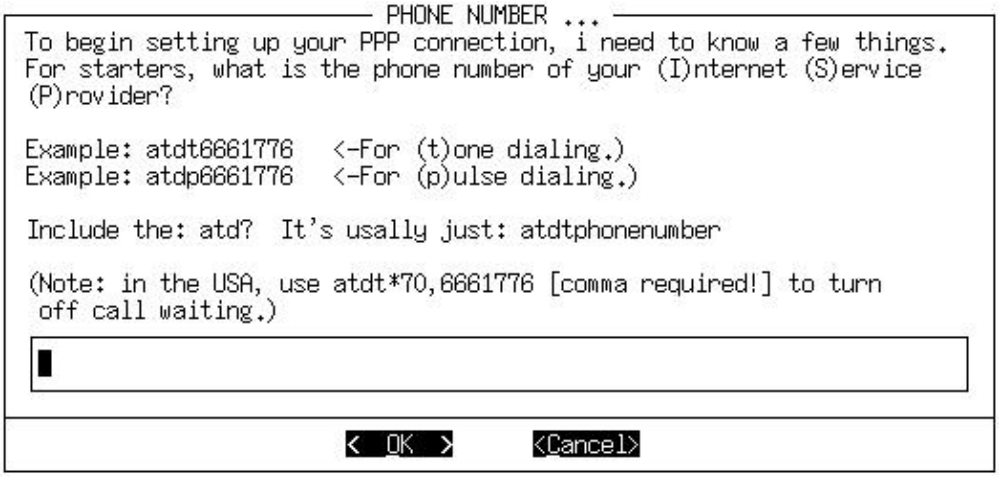
pppsetup

Slackware obsahuje utilitu **pppsetup** pro konfiguraci vytáčeného (dialup) spojení k ISP (Internet Service Provider – poskytovatel připojení k internetu). Najdete ji v balíčku `ppp.tgz` v softwarové skupině N. **pppsetup** se ovládá stejně jako program **setup**. Pokud si nepamatujete jak to bylo, podívejte se zpět do sekce *Program setup* v kapitole 3. Program **pppsetup** vám položí řadu otázek a nastaví několik konfiguračních souborů v adresáři `/etc/ppp`. Jako root spusťte **pppsetup** a my vás provedeme těmi otázkami.



Telefonní číslo

První otázka vás vyzývá k zadání telefonního čísla vaš eho ISP, jako "předponu" čísla uvedete typ vytáčení. Větš ina lidí bude chtít tónové vytáčení. Je–li telefonní číslo vaš eho ISP 555–1013 a použ íváte tónové vytáčení, měli byste zadat do dialogového okénka **atdt5551013**.



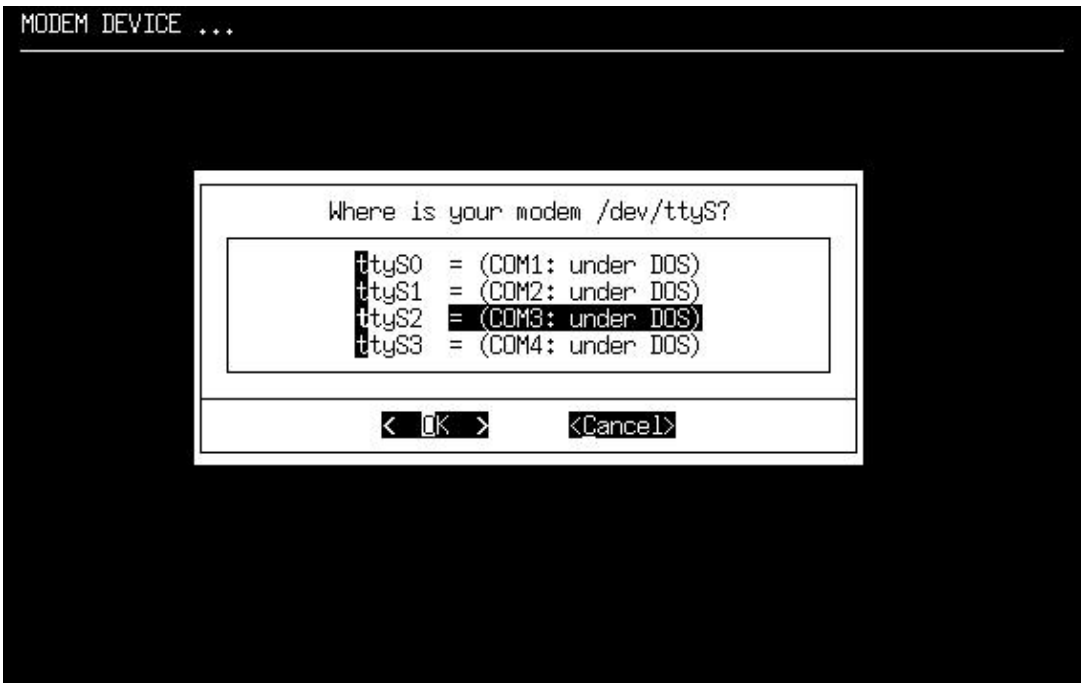
Máte–li čekání na oznamovací tón (call waiting) na vaš í telefonní lince a chcete ho vyřadit když se připojujete (obecně dobrý nápad), zadejte do dialogového boxu něco takového: **atdt*70,5551013**.

Ta čárka se vyž aduje. Tím dáváte 1,5 vteřinovou pauzu mezi *70 a číslo ISP, abyste vyřadili čekání na oznamovací tón. Bez té čárky to fungovat nebude.

pp: čekání na oznamovací tón lze rovněž vyřadit AT příkazem X3. Takž e obsah dialogového okna by mohl vypadat: **atx3dt5551013**.

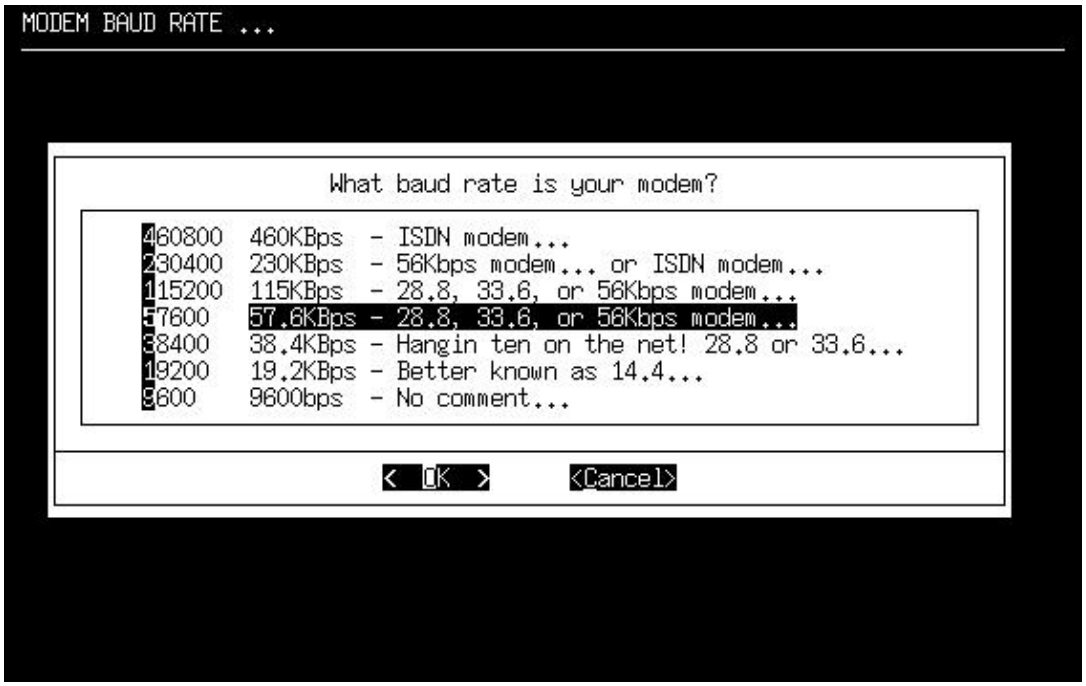
Modem – kde je

Dále vyberete umístění vaš eho modemu. Pokud víte jaké COM to bylo pod Windowsama, můž ete vybrat uvedený ekvivalent. Jinak mož ná budete muset trochu experimentovat. Nejlepš í bude začít na **ttyS0** a propracovávat se dolů seznamem.



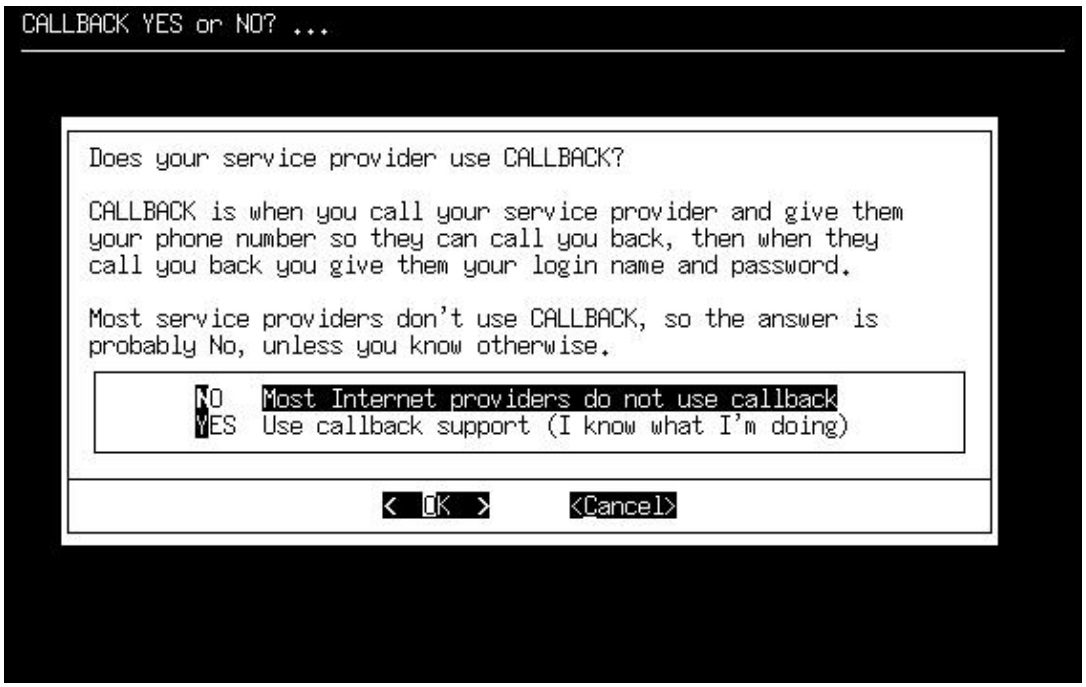
Rychlost modemu v baudech

Dále vyberte rychlost nejbližš í pro váš modem. Neznáte–li ji, měli byste si prohlédnout skříňku modemu nebo nějakou dokumentaci, kterou jste k němu dostali. Kaž dá volba má několik příkladů, takž e by nemělo být problém nastavit to.



Callback – zpětné zavolání

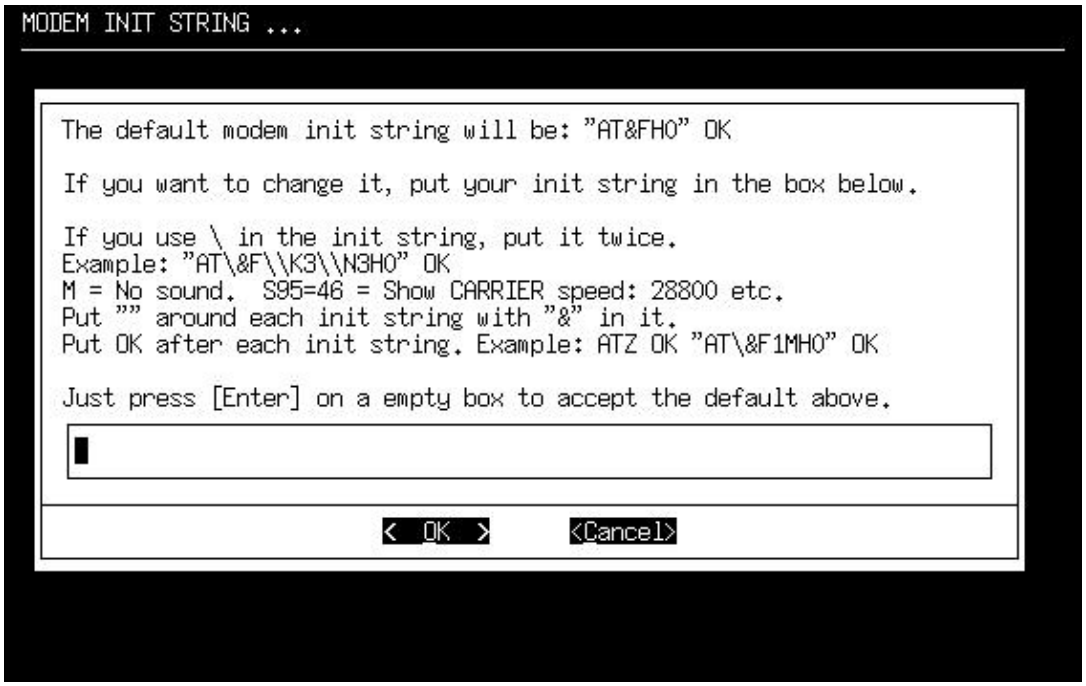
Teď byste se měli podívat do informací, které vám váš ISP poskytl. Nemnoho ISP používá callback, takže můžete nejspíš vybrat bezpečně "NO". Callback znamená, že když vytočíte číslo vašeho ISP, tak on vám zavolá zpět na vaše číslo a teprve pak se můžete přihlásit.



Musíte–li používat callback, vyberte "YES". Pak budete vyzváni k zadání vašeho telefonního čísla, přihlašovacího jména a hesla. Nemusíte zadávat úvodní přihlašovací jméno a heslo. Nakonec budete dotázáni, jaké ověřovací (authentication) schéma váš ISP používá. Používá–li CHAP nebo PAP, vyberete "YES". Později to ještě budete muset nastavit (viz sekce níže). Pokud ISP nepoužívá ani jeden z těch schémat, vyberte "NO" a nahlédněte do sekce "Chat script" níže.

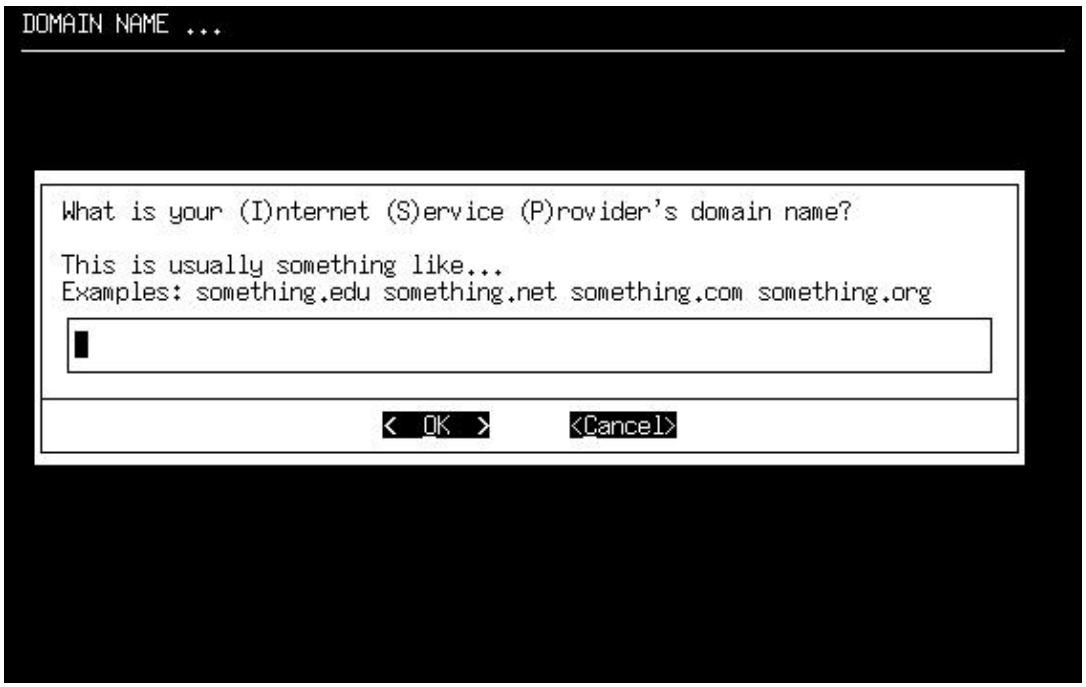
Inicializační řetězec pro modem (Modem init string)

Pokud nemáte nějaký obzvláštní modem, můžete nejspíš následující otázku jen odentrovat a tím vybrat defaultní inicializační řetězec (AT&FH0). V opačném případě se poraďte s dokumentací, kterou jste k vašemu modemu dostali, co se má jako inicializační skript používat.



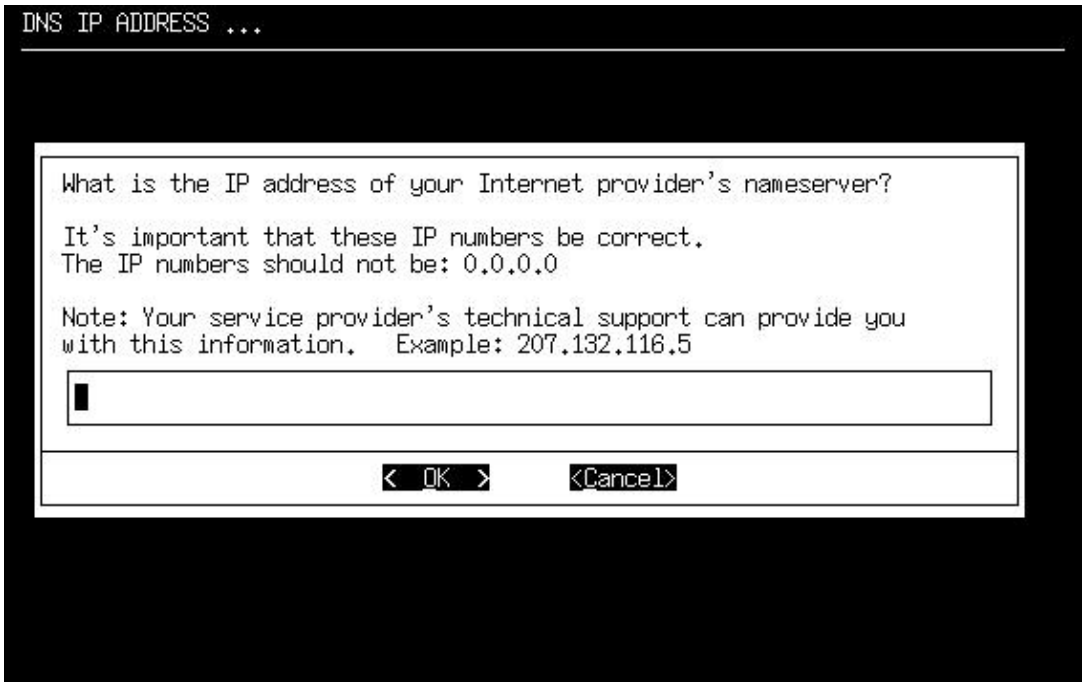
Jméno domény

Nyní zadáte doménové jméno vaš eho ISP. Bude to zhruba v této formě: example.net , slackware.com , nebo něco podobného. (No dobrá, nejspíš určitě to nebude slackware.com :o)



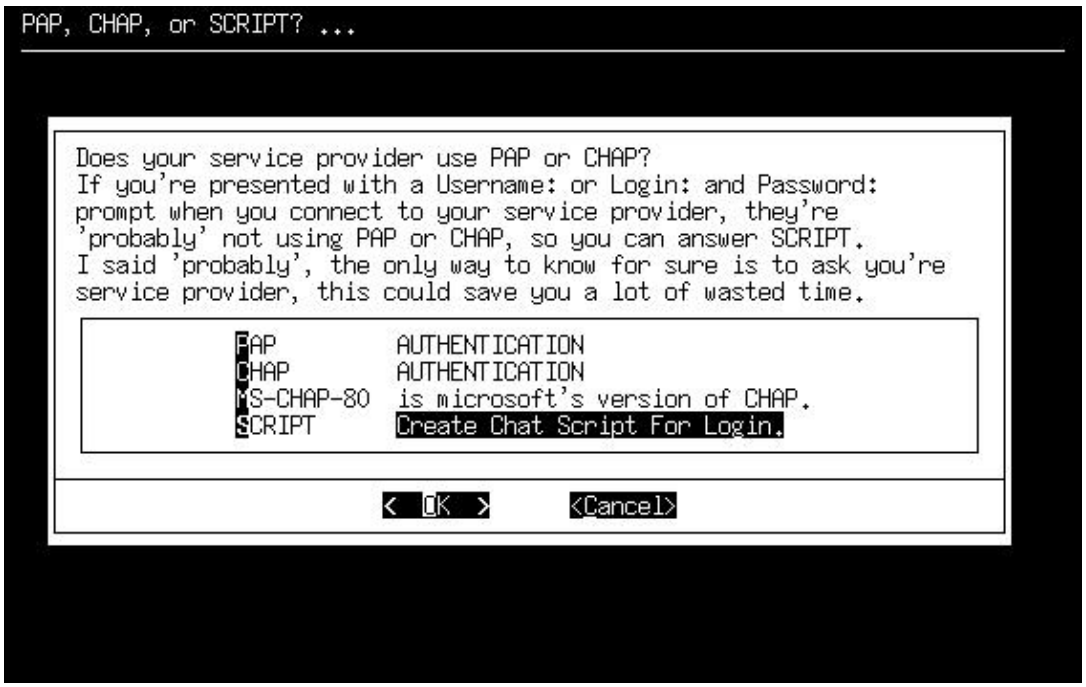
IP addresa DNS

Váš ISP by vám měl poskytnout i IP adresu jeho nameserveru (DNS). Pokud jste tuto adresu dostali, zapiš te ji do rámečku. V opačném případě se na ní přeptejte vaš ho ISP.



Metoda ověřovnání (Authentication method)

Tato otázka může přinést trochu známé metody pokusu a omylu. Potřebujete zadat, jestli váš ISP používá CHAP, PAP, nebo žádnou z těchto metod pro ověřování uživatele. Nejsnadnější způsob jak to zjistit je zavolat vaše mu ISP. Pokud jste při připojování obtěžováni výzvami k zadání přihlašovacího jména a hesla, měli byste nejspíše vybrat "SCRIPT". V opačném případě se poraďte s vaším ISP, jakou metodu používá.



PAP nebo CHAP

Pokud jste v obrazovce s ověřovací metodou vybrali "PAP" nebo "CHAP", budete teď muset zadat vaše uživatelské jméno. Váš ISP by vám ho měl přidělit. Pokud to neudělal, je tu něco hodně špatného. Budete muset vašeho ISP kontaktovat a vyžádat si to jméno od něho.

V následujícím dialogovém rámečku zadáte heslo, které vám váš ISP přidělil.

Chat skript

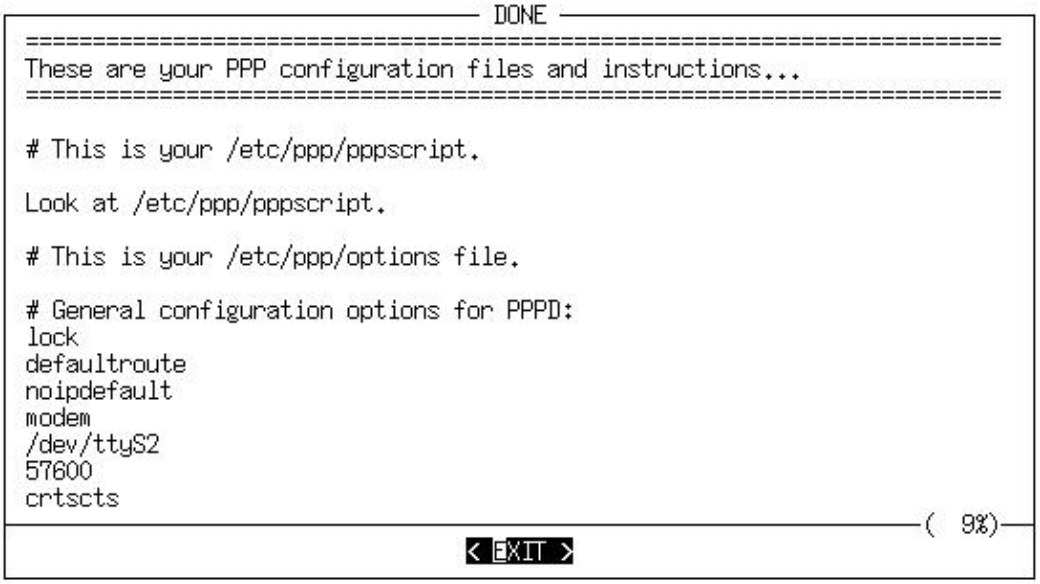
Pokud jste v obrazovce s ověřovací metodou vybrali "SCRIPT", budete obšťastněni opravdu dlouhou diskuzí na téma, co to ten chat script je. Pořádně si to přečtěte, protože popisuje všechno opravdu velice dobře. V podstatě tu budete muset zadat, jaké informace váš ISP bude posílat k vám a co by měl váš počítač posílat zpět (jako odpověď), aby vás přihlásili.

Ocitnete se v kolobě hu zadávání textů, které by váš počítač měl od vašeho ISP očekávat, a textů, které by měly být (jako odpověď) posílány zpět. Tento kolobě h můžete přeruš it stiskem klávesy enter při prázdném dialogovém rámečku.

Uděláno

Na závěr vám budou ukázány kompletní konfigurační soubory pro ppp. Nemůžete tu do nich nijak zasahovat, ale můžete si všechno přinejmenším zkontrolovat. Stiskněte enter, aby se všechno uložilo a **pppsetup** se ukončí.

Tato obrazovka vám rovněž podá množ ství informací o tom, jak navázat dialupové připojení a jak je zas přeruš it, když chcete skončit. Základní myš lenka je: Jako root spusť te **ppp−go** pro navázání spojení. Jakmile vám to předá místní a vzdálenou IP adresu, jste připojeni do internetu. Když budete končit, spusť e **ppp−off** (jako root), a spojení bude ukončeno.



Předchozí
Konfigurace sítě

Výchozí
Nahoru

Další
Soubory /etc

Soubory /etc

/etc/inetd.conf

V síťově orientovaných operačních systémech běží neobyčejně mnoho služeb. Typicky každá služba, která vyžaduje svůj program, potřebuje být spuštěná a naslouchat spojením. To se může stát velmi zatěžujícím v systémech, kde takových serverů běží hodně. Aby se toto omezilo, byl vytvořen inetd. Inetd je internet super–server –– je spuštěn a naslouchá spojením na mnoha soketech. Když nějaké přijde, inetd spustí příslušný server, který ho pak obslouží. Tím je množství čekajících serverů omezeno na jeden jediný.

Soubor `/etc/inetd.conf` je konfiguračním souborem pro inetd. Specifikuje, které servery se spouštějí pro které spojení. Manuálová stránka **inetd**(8) obsahuje mnohem detailnější informace, ale dovolme si aspoň krátké nahlédnutí na základní řádku služeb:

```
ftp  stream  tcp  nowait  root  /usr/sbin/tcpd  wu.ftpd -l -i -a
```

Toto je řádka pro ftp server. Povšimněte si, jak jméno protokolu ("ftp") na jedné straně a na druhé straně příkaz ke spuštění si odpovídají. V tomto případě program, který je spouštěn na základě pokusu o spojení je `/usr/sbin/tcpd`. To je "wrapper" (balící) program který poskytuje nějaké základní bezpečnostní volby pro server, který "obaluje". **wu.ftpd** je potom vlastním ftp serverem, ale nám ho spouští tcpd. Více informací o tcpd najdete v sekci *tcp_wrappers*.

(pozn.překl.: Proč to autoři komplikují? Normálně by tu nebyl tcpd, ale rovnou wuftpd. Jen z jistých bezpečnostních důvodů se ftp démon nespouští přímo ale je spouštěn prostřednictvím wrapperu tcpd)

Jako u mnoha systémových souborů i v `inetd.conf` jsou řádky komentářů uvozeny znakem #; můžete přidávat a vyjímat služby z inetd pouhým za– či od– komentářováním příslušné řádky a restartováním **inetd**.

/etc/resolv.conf

Tohle je soubor, který říká systému, kde sehnat DNS informace. Všechny nameservery, které používáte, by měly být uvedeny zde. Rovněž i jméno domény vašeho hostitelského počítače. Zde je příklad souboru `resolv.conf` (z laptopu na kterém tohle píšu, ninja.tdn):

```
domain tdn
nameserver 192.168.1.1
search tdn. slackware.com
```

První řádek uvádí jméno domény pro počítač "ninja"; to je vše, co je za hostname v mojí adrese. Druhý řádek uvádí DNS server pro naši domácí síť. Těch může mít tolik, kolik jich potřebujete; budou zkoušeny v pořadí od prvního po poslední kdykoliv nějaký program potřebuje vyhledat IP adresu odpovídající určitému doménovému jménu.

Poslední řádek je o něco zajímavější. Uvádí jakákoliv jména domén, která mají být považována z můj systém. Na příklad předpokládejme, že mám stroje zuul.tdn a hejaz.slackware.com. Můžu dát jednoduše **`ping zuul`** nebo **`ping hejaz`** a bude to fungovat. Jde to proto, že ping se nejdřív pokusí přidat ".tdn" ke jménu zuul, najde shodu a pokračuje s tímto jménem. V případě "hejaz" taky nejdřív zkusí "hejaz.tdn". Tady shodu nenajde, takže dále zkusí "hejaz.slackware.com"–– bingo. Povšimněte si, že doménová jména uvedená v řádku **`search`** musí být ukončena tečkou s výjimkou posledního. Je–li tu jediné doménové jméno, pak je posledním a ukončující tečku nepotřebuje.

/etc/hosts

Soubor `hosts` umožňuje nejjednodušší druh doménového vyhledávání. Je v něm seznam jmen hostitelů a jim odpovídající IP adresy. To je použitelné třeba u malých sítí, kde by DNS nebylo vhodné. Také v případě, kdy DNS server je nepolehlivý, atd. Rovněž je používán strojem v průběhu bootu, kdy ještě žádný name server není dostupný. Můj vypadá takhle:

```
127.0.0.1  localhost
192.168.1.32   ninja.tdn ninja
```

První řádek by měl být sebeozřejmující. Druhý možná ne. Můžete uvést tolik jmen a jejich aliasů pro danou adresu, kolik jen chcete. Oddělují se mezerou. Takže já mám "192.168.1.32" přeloženy na "ninja.tdn" (a naopak). Alias "ninja" mohu používat rovněž, když jsem moc líný psát to ".tdn" (což většinou jsem).

rc.inet1

`/etc/rc.d/rc.inet1` je souborem používaným k nastevení síťové "infrastruktury"–– inicializuje zařízení a nastavuje adresy a směrování. Soubor `/etc/rc.d/rc.inet1`, který je dodáván s Slackwarem je hezky zhusta okomentovaný, takže kdybychom ho tu chtěli popisovat, jen bychom opakovali sami sebe.

rc.inet2

Soubor `/etc/rc.d/rc.inet2` tu je pro další část síťové práce: Nastavuje služby a démony a obsluhuje mnohé zajímavé možnosti síťování. Podívejme se na příklad jednoho bloku:

```
# Start the NAMED/BIND name server:
if [ -f ${NET}/named ]; then
   echo -n " named"
   ${NET}/named -u daemon -g daemon
fi
```

Důležitou řádkou zde je ta čtvrtá, která spouští službu **named**(8). Zbytek je jen omáčka: Výraz "if" testuje, zda najde vlastní program **named** tam, kde očekává, že by měl být. Řádek **echo** podává zprávu, že **named** je spouštěn. Zjistíte, že většina serverů spouštěných skriptem `rc.inet2` je spouštěna z takovýchto bloků. Stručně: Nejdříve zkouší najít nějaký očividný důvod, proč službu nespustit, pak podají zprávu o spouštění a nakonec jsou příkazy, které spustí samotnou službu.

Rovněž `rc.inet2` je hezky zhusta okomentovaný; pohrabte se v něm trochu.

NFS (Network File System) – síťový souborový systém

NFS je obvykle používán ke sdílení souborů mezi různými počítači na síti. Skvělá věc na NFS je, že je navržen tak, aby si jeden stroj mohl připojit sdílené soubory z jiného stroje transparentně a přistupovat k nim stejně jako ke svým lokálním souborům.

Několik věcí se musí udělat, aby to fungovalo. První je, že na serverovém stroji musí být spuštěny odpovídající služby: Jsou to **portmap**(8), **nfsd**(8) a **mountd**(8). Druhá věc je, že server musí výslovně "exportovat" souborový strom pro konkrétního klienta. To se dělá prostřednictvím souboru `exports`(5) v adresáři `/etc`.

První část problému vyřešíme nainstalováním balíčku tcpip1.tgz (ze skupiny N) a necháme rc.inet2 dělat svoje věci. `/etc/exports` je o něco zábavnější.

Předpokládejme, že mám adresář s obrázky na počítači battlecat.tdn, který chci zpřístupnit pro počítač ninja.tdn. Na počítači battlecat budu potřebovat přidat do `/etc/exports` řádek:

```
/var/media/images   ninja.tdn(ro)
```

Pak na počítači ninja mohu jednoduše zadat

```
# mount -t nfs battlecat.tdn:/var/media/images /mnt
```

a adresář s obrázky se připojí pod adresář `/mnt`. Naneštěstí jsem si zakázal zápis do sdíleného adresáře–– ono "(ro)" v souboru `/etc/exports` na počítači battlecat je volba znamenající "read–only". Jakékoliv podobné volby musí být uvedeny v závorkách za jménem klientského počítače. Je–li jich více, oddělují se čárkami. Na příklad:

```
/var/media/images   ninja.tdn(rw,no_root_squash)
```

To "rw" je samozřejmě "read–write"–– *subject to user and group id mapping* (viz manuálová stránka `exports`(5), kde se to vysvětluje), uživatelé na počítači ninja mají dovoleno zapisovat do sdíleného adresáře. Je to celkem pohodlné, co?

tcp_wrappers

tcp_wrappers je základní systém pro zabránění (nebo výslovné povolení) přístupu ke službám ze specifikovaných hostitelských počítačů. V kostce to pracuje takto:

inetd (internet super−server) spouští mnoho různých serverů; mnoho jich je "wrapped" prostřednictvím **tcpd**. Jinými slovy, ty servery ve skutečnosti spouští **tcpd**, ale **inetd** to neví (a ani se o to nestará). **tcpd** zaznamená pokus o připojení a pak zkontroluje soubory `/etc/hosts.allow` a `/etc/hosts.deny`, aby se podíval, zda může být připojení povoleno.

Pravidla obsažená v těchto souborech mohou být poněkud komplexnější, ale předpokládejme, že pyramid.tdn je opravdu protivný a nenechá ubohého malého mojo.tdn napokoji. Tak mojo.tdn může vrhnout takovýto řádek do `/etc/hosts.deny`:

```
ALL: pyramid.tdn
```

Tento řádek by měl být dost jasný: Zabraňuje pyramidovi užívat na počítači mojo jakékoliv služby, které jsou chráněny promocí tcpd. Když budu obtěžován celou doménou (kromě té pyramidy), mohl bych řádek takto vylepšit:

```
ALL: pyramid.tdn, .otravna.domena
```

Jenže počkejme! Můj kámoš Hobbes sedí u počítače na .te.otravne.domene, ale já mu chci umožnit přístup ke mně (ostatním otravům ale ne). I to je snadné. Nechám už soubor hosts.deny tak jak je, a přidám následující řádek do hosts.allow, aby Hobbes mohl ke mně:

```
ALL: hobbes.otravna.domena
```

Mnohem více toho najdete v **tcpd**(8), v hosts_access(5) a v hosts_options(5). Systém tcp_wrappers je mnohem flexibilnější než jak jsme si ukázali a stojí za námahu zabývat se jím mnohem hlouběji.

Shrnutí

V této kapitole jste se naučili jak si nastavit systém tak, abyste se připojili do sítě, ukázali jsme si jak upravovat konfigurační soubory a řekli si o základních principech bezpečnosti. Dále jste se dozvěděli, co je to NFS (síťový souborový systém) a jak zařídit, aby na vašem systému pracoval. Tím, že se počítač stane součástí sítě, máte umožněn přístup ke všem druhům zdrojů jako je mail, news a webové stránky. Kapitola 13 popisuje, jak používat některé základní síťové programy.

XF86Setup

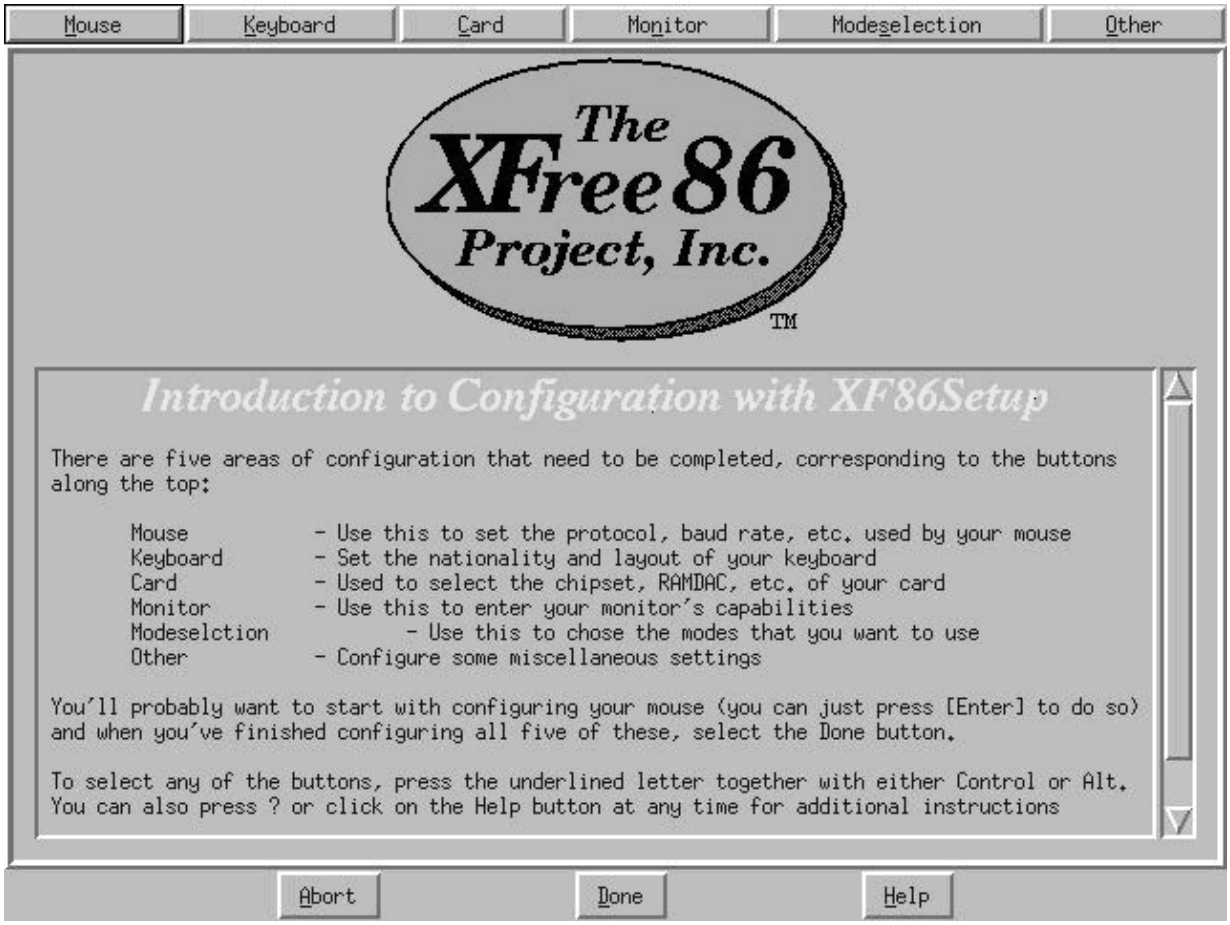
Druhá možnost jak nastavit X je použít **XF86Setup**. To je grafický konfigurační program, který je součástí balíčku `xset.tgz`. Rovněž budete potřebovat mít nainstalovaný balíček `xvg16.tgz`.

Abyste spustili XF86Setup, přihlaste se jako root a napište:

```
# XF86Setup
```

Pokud už máte vytvořený soubor `/etc/XF86Config` (po nějakém předchozím nastavování X), budete dotázáni, zda se nastavení v něm uvedená mají nyní použít jako výchozí. Pokud tento soubor ještě nemáte, dostanete se přímo do grafického módu.

Obrázek 6–1. Úvodní obrazovka programu `XF86Setup`.



Program **XF86Setup** je velmi podobný programu **xf86config**. Dotazuje se na ty samé věci, ale vše probíhá v grafickém prostředí. Nevíte–li si s nějakým dotazem rady, nahlédněte do předchozí kapitoly. **XF86Setup** má dost bohatou nápovědu, takže byste neměli mít žádné potíže při jeho vyplňování.

Poznámky překladatele: Přiznávám, že nevím jak přeložit "session". V daném kontextu to slovo znamená "spuštění" nebo "běh" systému X... Asi by se mi nejvíc zamlouvalo to "běh", běhové konfigurační soubory, ale kdo by tomu rozumněl? :o)) Máte-li nápad jak to překládat, dejte vědět. Na "výchozí" stránce – obsahu je na mě mail.
Dále, soubory, které jsou dle originálního textu umístěné pod adresářem /var/... jsou ve verzi Slackware 8 k nalezní pod /usr/... Podobných mírných nepřesností může být víc. Nemělo by nás to ale odradit.:o)

Konfigurační soubory pro běh (session) X

xinitrc a ~/.xinitrc

xinit(1) je program který provádí vlastní spuštění X; je volán programem **startx**(1), takže se jím nemusíte zabývat. Jeho konfigurační soubor určuje, jaké programy (včetně a zvláště okenního manažeru) se mají spustit při startu X. **xinit** nejdřív nahlédne do vašeho domovského adresáře, je-li tam soubor `.xinitrc`. Nalezne-li ho, spustí jej. Nenajde-li ho tam, spustí defaultní systémový `/var/X11R6/lib/xinit/xinitrc`. Tady je ukázka jednoduchého souboru xinitrc:

```
#!/bin/sh
# $XConsortium: xinitrc.cpp,v 1.4 91/08/22 11:41:34 rws Exp $

userresources=$HOME/.Xresources
usermodmap=$HOME/.Xmodmap
sysresources=/usr/X11R6/lib/X11/xinit/.Xresources
sysmodmap=/usr/X11R6/lib/X11/xinit/.Xmodmap

# merge in defaults and keymaps

if [ -f $sysresources ]; then
    xrdb -merge $sysresources
fi

if [ -f $sysmodmap ]; then
    xmodmap $sysmodmap
fi

if [ -f $userresources ]; then
    xrdb -merge $userresources
fi

if [ -f $usermodmap ]; then
    xmodmap $usermodmap
fi

# start some nice programs

twm &
xclock -geometry 50x50-1+1 &
xterm -geometry 80x50+494+51 &
xterm -geometry 80x20+494-0 &
exec xterm -geometry 80x66+0+0 -name login
```

Všechny ty "if" bloky tu jsou proto, aby se začlenily různá konfigurační nastavení z jiných souborů. Do souboru `.Xresources` se podíváme za chvíli, ale `.Xmodmap` necháme být. Zajímavá část vypsaného souboru xinitrc je na jeho konci, kde jsou spouštěny různé programy. Běh X bude zahájen správcem oken **twm**(1), hodinami a třemi terminály. Všimněte si **exec** před posledním terminálem. Tímto se nahradí aktuální běžící shell (ten, který zpracovává skript `xinitrc`) příkazem **xterm**(1). Když pak uživatel ukončí tento terminál, ukončí se tak i běh X.

Abyste si přizpůsobili startování X, zkopírujte si defaultní `/var/X11R6/lib/xinit/xinitrc` do `~/.xinitrc` a upravte si ho. Ty programové řádky na konci můžete nahradit čímkoliv se vám zlíbí. Konec mého .xinitrc je jednoduchý:

```
# Start the window manager:
exec startkde
```

Všimněte si, že v `/var/X11R6/lib/xinit` je několik souborů `xinitrc.*`, které korespondují s názvy různých správců oken a GUI. Můžete si používat kterýkoliv se vám zlíbí.

.Xresources a .Xdefaults

Mnoho programů pro X používá systém nazvaný "X Resource Database" (databáze zdrojů X), z něhož získávají různé preference uživatelů (fonty, barvy, atd.) Tato databáze je udržovaná programem **xrdb**(1), který pravděpodobně nikdy nebudete potřebovat spouštět přímo. V Slackware je místo toho spouštěn z `xinitrc`.
*The file that `xinitrc` tells **xrdb** to source for options is `~/.Xresources`.*
Soubor, který `xinitrc` žádá **xrdb** o zdroj pro volby je `~/.Xresources`.
***xrdb** will also load `~/.Xdefaults`, so either of these filenames will work. A minimal `.Xresources` file looks like this:*
xrdb bude rovněž načítat soubor `~/.Xdefaults`. Takže oba tyto názvy souborů budou pracovat. Minimální soubor `.Xresources` by mohl vypadat následovně:

```
xterm*background: black
xterm*foreground: gray
xterm*scrollBar: true
xterm*font: -*-lucidatypewriter-*-r-*-*-15-*-*-*-*-*-*-*
```

Tyto čtyři řádky specifikují konfigurační informace pro program **xterm**. X-ové zdroje se zapisují takto:

```
program*option: setting/value
```

Výše uvedený vzorový `.Xresources` by měl být dostatečně sebeozřejmující. Nenechte se rozhodit tím "font" řádkem; v X se fonty vždycky specifikují tímto způsobem.

Předchozí XF86Setup | Výchozí Nahoru | Další Servery a Window managery

Servery and Window managery

X window systém byl původně navržen tak, aby pracoval trasnparentně napříč sítí. Jeden velký server může sám spouštět X–ové programy, které pak mohou být zobrazeny na různých klientských strojích kdekoliv na síti. Schopnost vzdáleně zobrazovat programy může být velkou výhodou. Hlavní nevýhodou tohoto síťového konceptu je to, že je méně bezpečný, než když spouštíte aplikace na lokálním počítači, a dále že pro svou práci zabírá velkou část síťového pásma. Toto diskutujeme dále v kapitole *Exporting displays*.

I když spouštíte X–ka na vaš em vlastním stroji, stále se pohybujete v modelu klient/server. Serverem je ta část, která je specifická pro vaš i videokartu. Když jste nastavovali X a říkali jim, jaký druh videokarty máte, říkali jste jim vlastně, jaký druh serverového programu mají používat. Klientskou část představují všecičky ostatní programy, které si pod X spouštíte. Jeden speciální klient, nazývaný window manager (správce oken), zodpovídá za vzhled a styl vaš ich X. Okenního správce probereme detailněji později.

Prací správce oken je obsluhovat vykreslování oken s programy na obrazovce, a rovněž obsluhovat vstup z myš i a klávesnice. První manažery dělali téměř jenom tohle a nic moc více. Dnešní okenní správci jsou daleko komplikovanější a jsou přizpůsobovatelné a nastavovatelné jakýmkoliv představitelným způsobem. Nabízejí váš í fantazii spoustu možností, které vám umožní mít vzhled desktopu odlišný od kohokoliv jiného.

Množství okenních manažerů znatelně odlišuje desktopy Linuxu od Windows. Pod Windows máte jedno základní okenní prostředí. Pod Linuxem můžete spouštět kterýkoliv z mnoha okenních správců; každý s odlišným vzhledem, jinými rysy a možnostmi. Někteří lidé by to označili za slabinu, protože tu není konzistetní vzhed. Nicméně většina uživatelů Linuxu to považuje za přednost, protože si můžete nakonfigurovat systém zcela tak, jak chcete.

Vybíráme si desktop

Po léta byl Unix téměř výhradně používán jako operační systém pro servery s výjimkou vysoce−výkonných pracovních stanic. Jenom technicky zaměření lidé měli v oblibě používání Unixových operačních systémů a uživatelský interface tento fakt odrážel. GUI bylo velmi strohé, navržené ke spouštění několika nezbytně grafických aplikací jako jsou CAD programy a vykreslovače obrázků. Většina souborů a správy systému byla ovládána z příkazové řádky. Různí dodavatelé (Sun Microsystems, Silicon Graphics, atd.) prodávali instalace pracovních stanic s pokusem o soudržný vzhled a styl, ale široká škála grafických toolkitů, které používali vývojáři vedla nevyhnutelně k rozbití snahy o jednotnost desktopu. Už jen rolovací posuvník nemusel vypadat u dvou aplikací stejně. Menu se mohla objevovat na různých místech. Programy měly různá tlačítka a zaškrtávací políčka. Barvy byly různých palet a byly obvykle generovány natvrdo v každém toolkitu. A že uživateli byli především těmi techničtí profesionálové, nikomu to příliš nevadilo.

S příchodem volných Unixových operačních systémů a zvyšujícím se počtem a rozličností grafických aplikací, získala nedávno X širokou uživatelskou základnu. Většina uživatelů byla pochopitelně zvyklá na konzistentní vzhled a styl nabízený Microsoftími Windows nebo Applovským MacOS. Nedostatek této konzistentnosti v X−ových aplikacích se stal překážkou jejich širšího přijetí. V reakci na to byly zahájeny dva open source projekty: K Desktop Environment, neboli KDE a GNU Network Object Model Environment, známý jako GNOME. Oba mají širokou škálu aplikací od "taskbars" a souborových manažerů až po hry a kancelářské balíky, napsané s tím samý grafickým toolkitem a těsně integrované, aby poskytovaly jednotný, konzistentní desktop.

Rozdíly mezi KDE a GNOME jsou celkem malé. Nabízejí odlišný vzhled, protože používají různé grafické toolkity. KDE je založeno na knihovně Qt od Troll Tech AS, zatímco GNOME používá GTK – toolkit původně vyvinutý pro GNU Image Manipulation Program (zkráceně GIMP). Jako oddělené projekty mají KDE a GNOME své vlastní vývojáře a programátory s odlišnými vývojářskými styly a filozofií. Výsledek je v každém případě, přeze všechno fundamentálně stejný: Konzistentní, těsně integrované prostředí desktopu a sbírka aplikací. Funkčnost, využitelnost a čistá krása obou rivalů nabízí vše, co je dostupné na jiných operačních systémech.

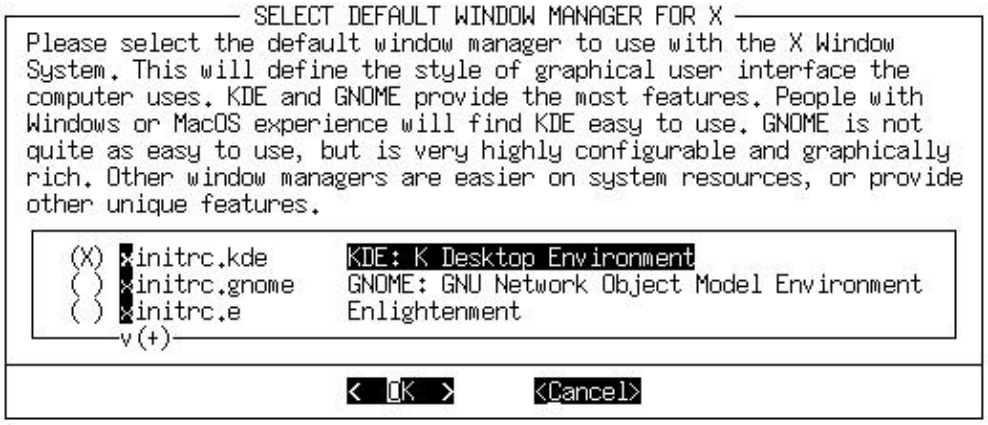
Nejlepším na tom je, že oba vyvinuté dektopy jsou volně zdarma. To znamená, že můžete mít jeden nebo oba (ano najednou). Volba je na vás.

Kromě desktopů GNOME a KDE, obsahuje Slackware širokou škálu dalších okenních manažerů. Některé jsou navrženy tak, aby emulovaly jiné operační systémy, jiné tak, aby byly hodně přizpůsobivé uživateli, jiné tak, aby byly rychlé. V pohodě si jich můžete naistalovat tolik, kolik chcete, pohrát si s nimi všemi a rozhodnout se, který si oblíbíte nejvíce.

Aby bylo vybírání desktopu snadné, Slackware obsahuje program **xwmconfig**, který můžete použít k výběru desktopu nebo(li) správce oken. Spouští se takto:

```
$ xwmconfig
```

Obrázek 6−2. Slackware `xwmconfig` program.



Je vám nabídnut seznam všech desktopů a správců oken, které jsou nainstalovány. Prostě si jen ze seznamu vyberte ten, který chcete. Každý uživatel na vašem počítači si bude muset spustit tento program, protože různí uživatelé mohou používat různé desktopy a ne každý bude chtít ten defaultní, který jste vybrali při instalaci.

Potom jen nastartujte X a můžete jet:

```
$ startx
```

Exporting displays

As was previously mentioned, it is possible to run X programs on one computer and display them on another. This is incredibly bandwidth–intensive, so you probably won't want to do this over a modem connection or over very long distances. Additionally, there are security considerations: exporting a display is not a very secure thing to do, since you'll be letting the entire network look at what you're doing. Still, it can be very useful on a local network.

An important thing to note here is the use of the words client and server . When exporting the display, you may become confused as to what's a client and what's a server. We refer to the machine that actually runs the X programs and sends the display information as the server . The machine that you use to display the remote program is called the client . When discussing the design of X, this is reversed. The program that displays things is called the server , while the running program is called the client . It's not too terribly confusing, but worth pointing out.

For this example, we'll be making use of two computers: golf is a fairly powerful server sitting underneath a desk on one side of a crowded room. It has a lot of RAM and a nice processor in it. In addition, it has lots of X programs on it, but no monitor. On the other side of the room is couch, an old machine with little RAM and not much disk. It is way too weak to run resource–intensive programs like Netscape. couch has two major advantages, though: it has a monitor, and it is sitting right next to the couch so you don't even have to get up to use it. Ideally, you would be able to run Netscape without getting off the couch. Exporting is the answer.

First, log in to couch and start up X. Then open your favorite terminal program (xterm, rxvt, eterm, aterm, or a host of others). The first step to remotely displaying X programs is to set up the client machine so that other machines are allowed to display to the machine. This uses the **xhost** program to control access. If you are on a secure internal network, you probably don't care who can remotely display programs. In that case, you would just let anyone on the network display:

```
couch$ xhost +
access control disabled, clients can connect from any host
```

On the other hand, you might want to do this using machines that are on an insecure network (the Internet, a college network, or anything else that you don't have control over). You certainly don't want just anyone to connect. xhost allows you to be selective about who can display:

```
couch$ xhost + golf.foc
golf.foc being added to access control list
```

Now, only golf.foc (the server mentioned earlier) can display programs to couch. You can see who has access to display programs by running xhost with no arguments:

```
couch$ xhost
access control enabled, only authorized clients can connect
INET:golf.foc
INET:localhost
INET:couch.foc
LOCAL:
```

This is all you have to do to set up the client end of things. The next step is to set up the server so that it knows to display programs somewhere other than the monitor. Since the server doesn't have a monitor on it (and therefore doesn't have X running), it'll need to know where to display.

Setting up the server is not very difficult either. After connecting, you'll need to modify the `$DISPLAY` environment variable. By default, it will probably not be set to anything. You'll need to set `$DISPLAY` to the value of the remote host, plus a number that represents which X session to display to. You will almost always have only one X session running, so dealing with that variable shouldn't be an issue.

Here's how the `$DISPLAY` variable would be set on our example server using Bash as the shell. Other shells would use a different syntax, but the value should be the same.

```
golf$ export DISPLAY=couch.foc:0.0
```

That's all there is to setting up the server side of things. Now you just stay logged into the server and run X programs from there. All the screen output from the program will get sent across the network to the client machine, even though it's running on a computer across the room.

```
golf$ netscape &
```

This would run **netscape** off the server machine, but since the `DISPLAY` variable is set to couch, everything will get displayed there. You don't even have to get up to run those big X programs on your old terminal. One important note about this: the server machine will have to have all the X libraries and other support files needed to run the program. However, you won't need an X server or a `/etc/XF86Config` file, since nothing is getting displayed to the server.

Afterwards, you might want to disable display exporting by removing the server from your client's access control list:

```
couch$ xhost - golf.foc
golf.foc being removed from access control list
couch$
```

You can see how this a great way to share computing resources. But be careful, you may be the host of many X programs for many remote computers and not even know it.

Shrnutí

V této kapitole jste se naučili, jak si s pomocí programů **xf86config** a **XF86Setup** nakonfigurovat X Window systém. Také byste měli vědět, co to je prostředí desktopu a správce oken (window manager) a jak se přepínat mezi různými možnostmi. Měli byste být schopni exportovat session X na jiný počítač. V tomto bodě byste měli být připraveni spoušt ět a provozovat grafické prostředí.

LOADLIN

Slackware Linux nabízí ješ tě dalš í mož nost jak bootovat a tou je LOADLIN. LOADLIN je spustitelný program pro DOS, který můž ete použ ít ke spuš tění Linuxu přímo z běž ícího DOSu. Vyž aduje, aby na DOSovském oddílu byl linuxový kernel, aby jej LOADLIN mohl zavést a systém řádně nabootovat.

V průběhu instalační procedury je LOADLIN nakopírován do rootova domovského adresáře jako .zip soubor. Neexistuje automatický nastavovací proces pro LOADLIN. Budete potřebovat zkopírovat linuxové jádro (`/vmlinuz`) a soubor LOADLIN z rootova adresáře na DOSový oddíl.

LOADLIN se dá výhodně použ ít na vytvořitení bootovací menu na vaš em DOSovském oddílu. Do souboru `AUTOEXEC.BAT` můž ete přidat menu, které vám umož ní vybírat mezi Linuxem a DOSem. Zvolením Linuxu se spustí LOADLIN, který nabootuje váš Slackware systém. Dále uvedený `AUTOEXEC.BAT` pro Windows95 nabízí vyhovující bootovací menu:

```
@ECHO OFF
SET PROMPT=$P$G
SET PATH=C:\WINDOWS;C:\WINDOWS\COMMAND;C:\
CLS
ECHO Please Select Your Operating System:
ECHO.
ECHO [1] Slackware Linux
ECHO [2] Windows 95
ECHO.
CHOICE /C:12 "Selection? -> "
IF ERRORLEVEL 2 GOTO WIN
IF ERRORLEVEL 1 GOTO LINUX
:WIN
CLS
ECHO Starting Windows 95...
WIN
GOTO END
:LINUX
ECHO Starting Slackware Linux...
CD \LINUX
LOADLIN C:\LINUX\VMLINUZ ROOT=<root partition device> RO
GOTO END
:END
```

Musíte specifikovat, kde máte root oddíl (root partition device) ve tvaru v jakém se zadávají v Linuxu jména zařízení (třeba `/dev/hda2`).

Rovněž můž ete spouš tět LOADLIN z příkazové řádky. Napíš ete totéž , co je uvedeno v předchozím příkladě. V dokumentaci k LOADLINu je množ ství příkladů, jak jej použ ívat

Dual Booting

Mnoho uživatelů nastavuje svoje počítače tak, aby mohli bootovat Slackware Linux a ještě další operační systém. V následujícím textu jsme popsali několik typických scénářů pro dual boot, abyste neměli zbytečné problémy při nastavování vašeho systému.

Windows 9x/DOS

Nastavování počítače se systémem Windows 9x a Linuxem je asi ten nejobvyklejší dual bootovací scénář. Existuje množství způsobů jak můžete nastavit bootování; my zde popíšeme dva.

Člověk častokrát, když nastavuje dual bootový systém, vymyslí dokonalý plán kam všechno přijde, ale zachybuje v pořadí instalačních kroků. Je důležité pochopit, že operační systémy musí být nainstalovány v jistém pořadí, aby dual boot fungoval. Linux vždy poskytuje kontrolu nad tím, co (pokud něco) bude zapsáno do Master Boot Record (MBR). Proto je vždy lepší instalovat Linux jako poslední. Windows by měly být nainstalovány dříve, protože ony zapisují svůj zaváděcí program do MBR vždy (a bez ptaní).

Použití LILO

Většina lidí si bude přát používat k výběru mezi Linuxem a Windows LILO. Jak zmíněno výše, měli byste nejdřív nainstalovat Windows a teprve potom Linux.

Řekněme, že máte 47GB velký IDE harddisk, který je jedinou diskovou jednotku ve vašem počítači. Dále řekněme, že chcete dát půl diskového prostoru na Windows a polovinu Linuxu. To bude pro bootování Linuxu představovat problém. I když neznám specifickou geometrii toho disku, je téměř jisté, že druhá polovina disku bude začínat pěkně daleko za 1024 cylindrem. Lepší rozvržení pro tento disk by bylo:

```
1GB   Windows boot (C:)
1GB   Linux root (/)
22.5  Windows misc (D:)
22.5  Linux /usr (/usr)
```

Mimo to, budete chtít asi taky vyhradit adekvátní prostor na disku pro linuxový swap oddíl. Nepsaným pravidlem je, že byste měli pro swap vyhradit na disku prostor o velikosti dvojnásobku velikosti RAM. 64MB systém by měl mít 128MB swap, ap.

Po vytvoření diskových oddílů byste měli pokračovat instalací Windows. Až ty budete mít nastaveny a funkční, měli byste nainstalovat Linux. Instalace LILO vyžaduje zvláštní pozornost. Pro jeho nainstalováni byste měli zvolit mód "expert" (v programu liloconfig).

Začněme novou konfiguraci LILO. Budeme ho chtít nainstalovat do MBR, aby mohlo být využíváno k vybírání mezi oběma operačními systémy. Z menu, které vidíte, přidejte linuxový oddíl a Windows (či DOS) oddíl. Až to budete mít hotovo, můžete nainstalovat LILO.

Restartujte počítač. LILO by se mělo spustit a čekat na zásah uživatele. Můžete stisknout **Alt**, abyste získali `boot:` prompt. Napište jméno operačního systému který chcete zavést (tato jména byla vybrána, když jste nastavovali LILO). Pokud jste je zapomněli, stiskněte **Tab** a seznam jmen operačních systémů se vypíše.

LILO můžete rovněž nakonfigurovat ručním editováním souboru `/etc/lilo.conf` (je v linuxovém oddílu). Můžete je nastavit tak, aby se zobrazovalo textové menu a aby byl vždy zobrazen prompt. Kdybych chtěl, aby mi LILO zobrazovalo tohle:

```
System Boot Menu
================
1 - Linux
2 - Windows

LILO boot:
```

Moje `/etc/lilo.conf` by mělo vypadat nějak takhle:

```
# LILO configuration file

boot = /dev/hda
vga = normal
message = /boot/message

image = /vmlinuz
   root = /dev/hda2
   label = 1
   read-only

other = /dev/hda1
   label = 2
   table = /dev/hda
```

A můj soubor `/boot/message` by měl vypadat takto:

```
System Boot Menu
================
1 - Linux
2 - Windows
```

LILO je fakt dobře konfigurovatelný zavaděč. Není omezen jen na bootování Linuxu či DOSu. Můžete bootovat prakticky cokoliv. Manuálové stránky pro **lilo**(8) a **lilo.conf**(5) poskytnou mnohem podrobnější informace.

Ale co když LILO nefunguje? Stává se, že na některých strojích LILO nepracuje. Naštěstí tu máme ještě jiný způsob pro dual boot Linuxu a Window.

Použití LOADLIN

Tato metoda se používá, když LILO na vaším systému nefunguje, nebo když se vám jenom nechce LILO nastavovat. Tato metoda je rovněž ideální pro ty uživatele, kteří svá Windows často přeinstalovávají. Při každé reinstalaci vám Windows přepíší MBR a tím zničí nainstalované LILO. S LOADLINem tomuto problému nepodléháte. Největší nevýhodou je, že můžete používat pouze LOADLIN k bootování Linuxu.

Pro LOADLIN můžete instalovat operační systémy v jakémkoliv pořadí. Dejte si pozor při instalování věcí do MBR, abyste tam něco nezapsali omylem. LOADLIN je závislý na tom, aby windowsový oddíl byl bootovatelný. Takže při instalaci Slackwaru pozorně přeskočte nastavování LILO.

Po nainstalování operačních systémů zkopírujte soubor `lodlinX.zip` (kde "X" je číslo verze, např. "16a") z rootova domovského adresáře na vaš Windows oddíl. Taky tam zkopírujte obraz jádra. Musíte být v Linuxu, abyste tohle mohli udělat *(pozn.překl: Tak teď už to nevydržím, už to musím říct: Tenhle návod má chybu jak prase! Když bude v MBR zavaděč Windows a nebudete mít ani bootvací disketu, tak se do Linuxu prostě nedostanete! Nenabootujete tam a navíc z Windows není do linuxových oddílů vidět.)*. Tento příklad ukazuje, jak to uděláte:

```
# mkdir /win
# mount -t vfat /dev/hda1 /win
# mkdir /win/linux
# cd /root
# cp loadlin* /win/linux
# cp /vmlinuz /win/linux
# cd /win/linux
# unzip loadlin16a.zip
```

Takto vytvoříte adresář `C:\LINUX` na Windows oddílu (Předpokládáme, že je to `/dev/hda1`) a překopírujete nezbytné věci pro LOADLIN. Poté budete muset rebootovat do Windows, abyste tam nastavili bootovací menu.

Až budete ve Windows, dostaňte se do DOS promptu. Nejdříve se musíme ujistit, zda systém není nastaven tak, aby bootoval do grafického rozhraní.

```
C:\>cd \
C:\>attrib -r -a -s -h MSDOS.SYS
C:\>edit MSDOS.SYS
```

Přidejte do tohoto souboru tuto řádku:

```
BootGUI=0
```

Nyní soubor uložte a ukončete editor. Pak editujte `C:\AUTOEXEC.BAT` a přidejte tam bootovací menu, jak je ukázáno v tomto příkladě:

```
cls
echo System Boot Menu
echo.
echo 1 - Linux
echo 2 - Windows
echo.
choice /c:12 "Selection? -> "
if errorlevel 2 goto WIN
if errorlevel 1 goto LINUX
:LINUX
cls
echo "Starting Linux..."
cd \linux
loadlin c:\linux\vmlinuz root=/dev/hda2 ro
goto END
:WIN
cls
echo "Starting Windows..."
win
goto END
:END
```

Klíčovým je řádek, který spouští LOADLIN. Ten říká aby se zavedl kernel, udává na kterém oddílu je Linux, a že chceme, aby se v počátku připojil jen pro čtení (ro – read–only).

Slackware Linux poskytuje nástroje pro obě tyto metody. Na trhu je mnoho dalších zavaděčů (booter), ale tyto dva by měli pracovat ve většině dual bootových nastavení.

Windows NT

Toto je druhá nejčastější situace pro dual boot. Windows NT představují o něco více problémů, než dual boot mezi Win9x a Linuxem. Ten který nás znepokojuje nejvíc je, že když LILO přepíše MBR, nebudou NT bootovat správně. Proto musíme použít zavaděč OS, který je dodáván společně s WinNT. Následující kroky ukazují, jak byste měli nastavit dual boot systém pro WinNT a Linux.

1. Nainstalujte Windows NT
2. Nainstalujte Linux a dejte pozor, abyste LILO nainstalovali do superbloku linuxového oddílu.
3. Vezměte prvních 512 bytů linuxového root oddílu a uložte je do WinNT oddílu.
4. Editujte `C:\BOOT.INI` pod Windows NT a přidejte tam volbu pro Linux.

Instalace Windows NT by měla být celkem jasná, stejně tak instalace Linuxu. Ale potom už to začíná být zajímavé. Stáhnout prvních 512 bajtů linuxového oddílu je snazší než se zdá. Abyste to dokázali, musíte být v Linuxu. Předpokládejme, že váš linuxový oddíl je `/dev/hda2`. Zadejte tento příkaz:

```
# dd if=/dev/hda2 of=/tmp/bootsect.lnx bs=1 count=512
```

A je to. Teď potřebujete zkopírovat `bootsect.lnx` do oddílu s Windows NT. A tady se dostaneme do dalšího problému. Linux nemá spolehlivou podporu zápisu do souborového systému NTFS. Jestliže jste zformátovali oddíl pro instalaci Windows NT jako NTFS, budete muset náš souborek zkopírovat na FAT disketu, a potom ho z ní přečíst pod Windows NT. Pokud byste zformátovali oddíl pro WinNT jako FAT, pak byste ho mohli jednoduše namountovat pod Linux a soubor tam překopírovat. Tak či tak, budete potřebovat dostat soubor `/tmp/bootsect.lnx` z Linuxového

oddílu do `C:\BOOTSECT.LNX` na WinNT oddílu.

Posledním krokem je přidání položky menu do bootovací nabídky Windows NT. Pod Windows NT otevřte příkazový řádek.

```
C:\WINNT>cd \
C:\>attrib -r -a -s -h boot.ini
C:\>edit boot.ini
```

Přidejte tento řádek na konec tohoto souboru:

```
C:\bootsect.lnx="Slackware Linux"
```

Uložte změny, zavřete editor. Až budete rebootovat Windows NT, budete mít v menu volbu pro Linux. Jejím vybráním nabootujete do Linuxu.

Linux

Ano, lidé tohle opravdu dělají. Je to úplně ten nejjednodušší scénář pro dual boot. Jednoduše můžete použít LILO a přidat více položek do konfiguračního souboru `/etc/lilo.conf`. A to úplně stačí.

Předchozí LOADLIN | Výchozí Nahoru | Další Shrnutí

Shrnutí

V této kapitole jsme pojednali o bootování vaš eho systému s využ itím buď LILO, nebo Loadlin. Rovněž jsme probrali bootování mezi Linuxem a jinými operačními systémy. Měli byste být schopni správně zkonfigurovat vaš i bootovací metodu, stejně jako vybrat správnou metodu pro dual boot s dalš ím operačním systémem.

Příkazová řádka

Spouštění programů

Je dost těžké stát se úspěšným bez spouštění programů. Možná můžete svým počítačem něco podepřít, nebo jím udržovat dveře otevřené a něco bude vydávat ten nejmilovanější bzukot, když to poběží. Tak to opravdu jde. Jenže já si mysím, že všichni budou souhlasit, že využití počítače jako bzučící zarážky dveří není to, co by osobním počítačům přineslo tu oblíbenost, které se těší.

Takže, vzpomínáte si jak téměř vše v Linuxu ja souborem? Ano? Tak pro programy to platí taky. Každý příkaz, který zadáváte (není–li zabudován v shellu) sídlí někde jako soubor. Programy spouštíte jednoduše tak, že zadáte plnou cestu k tomuto souboru.

Například: Vzpomenete si na onen příkaz **su** z minulé kapitoly? No, tak ten je vlastně v adresáři `/bin` a zadání **/bin/su** ho hezky spustí.

No ale, jaktože to fungovalo, když jsme zadali jenom **su**? Skutečně jsme neřekli, že to je v `/bin`. Klidně to mohlo být v `/usr/local/share`, že jo? Tak jak to *ví*? Odpověď najdeme v proměnné prostředí `PATH` (cesta). Většina shellů má svou `PATH`, nebo něco podobného. Ta obsahuje seznam adresářů, které se mají prohledat, zda je tam příkaz, který se pokoušíte spustit. Takže, když spouštíte **su**, tak váš shell projde svůj seznam adresářů, aby zjistil, v kterém z nich se nachází spustitelný soubor jména **su**, který by mohl spustit. První odpovídající, který najde pak spustí. To se děje pokaždé, když spouštíte program bez toho, že byste uvedli úplnou cestu k němu. Pokud obdržíte chybovou hlášku `"Command not found"` (příkaz nenalezen), tak to znamená jen to, že program, který jste se pokusili spustit není ve vaší cestě (PATH). No, taky to může nastat, když ten program neexistuje vůbec. O proměnných prostředí se budeme bavit detailněji v kapitole *Bourne Again Shell (bash)*.

Pamatujte si, že zápis "." je kratší, než "/adrsář/kde/jsem", takže stane–li se, že jste v adresáři `/bin`, pak zápis **./su** bude pracovat stejně jako explicitní uvedení celé cesty (/bin/su).
(pp: Zkrátka "." je odvolávka na adresář, v němž se právě nalézáte.)

Porovnávání žolíků (wildcard)

Snad každý shell rozpoznává určité znaky jako náhražky či zkratky s významem "sem se hodí cokoliv". Takové znaky se nazývají případně "wildcards" (žolíky). Nejobvyklejšími jsou * a **?**. Je zažité, že **?** obyčejně nahrazuje jakýkoliv jeden znak. Například předpokládejme, že jste v adresáři, kde se nalézají tři soubory: `example1.txt`, `example2.txt` a `example3.txt`. Budete chtít zkopírovat všechny tyto soubory (příkazem **cp**, který popisujeme v sekci *cp* v 10. kapitole) do jiného adresáře, řekněme `/tmp`. Můžeme použít zápis **cp example1.txt example2.txt example3.txt /tmp**. Ale je snazší napsat **cp example?.txt /tmp**. Znak **?** se hodí místo kteréhokoliv ze znaků "1", "2" a "3" a v každém kole jím bude nahrazen.

Co tomu říkáte? Že to je *pořád* moc práce? Máte pravdu. Je to strašné. Máme přece zákoník práce, aby nás před takovou dřinou chránil. Naš těstí máme ještě *****. Jak už bylo zmíněno, * nahrazuje "jakékoliv množství libovolných znaků" (i žádný znak). Takže kdyby ty tři soubory byly v onom adresáři jedinými, mohli bychom jednoduše napsat **cp * /tmp** a dostat je všechny jednou ranou. Předpokládejme dále, že je tu ještě soubor `example.txt` a soubor `hejaz.txt`. Chceme zkopírovat `example.txt`, ale ne už `hejaz.txt`. `cp example* /tmp` nám to udělá.

cp example?.txt /tmp by dostal jen naše původní tři soubory, protože v `example.txt` není žádný znak, který by se místo **?** dosadil.

Přesměrování vstupu/výstupu a roury

(Teď přichází něco bezva.)

```
$ ps > blargh
```

Víte co to je? To jsem já, když spouštím **ps**, abych viděl, které procesy právě běží. Příkaz **ps** je popsán v 11. kapitole. To ještě nebyla ta bezva věc. Tou bezva věcí je **`> blargh`**, které znamená "vezmi výstup z ps a zapiš jej do souboru jména blargh". A počkejte, bude to ještě bezvadnější.

```
$ ps | less
```

To vezme výstup z **ps** a pomocí roury (pipe) jej předá příkazu less, který mi umožní, abych tím výpisem mohl rolovat nahoru a dolů jak chci.

```
$ ps >> blargh
```

Toto je třetí nejpoužívanější přesměrování. Dělá totéž jako ">", s tím rozdílem, že ">>" připojí výstup z **ps** na konec souboru `blargh`, když už existuje. Pokud ještě neexistuje, tak stejně jako v případě ">" bude vytvořen. (">" přepíše (nahradí) současný obsah souboru `blargh`.)

Ještě je tu operátor "<", který říká: "vezmi si jako vstup následující", ale to se nepoužívá tak často.

```
$ fromdos < dosfile.txt > unixfile.txt
```

Přesměrování se stává opravdovou zábavou ve chvíli, kdy je začnete kupit:

```
$ ps | tac >> blargh
```

To spustí **ps**, otočí pořadí řádků ve výstupu (tac) a připojí je na konec souboru `blargh`. Přesměrování můžete za sebe naskládat tolik, kolik chcete. Jen si zapamatujte, že jsou vyhodnocována zleva doprava.

V manuálových stránkách **bash**(1) najdete více informací o přesměrovávání.

Bourne Again Shell (bash)

Proměnné prostředí

Linux je složitá bestie a je tu toho hodně, co ji drží v mezích. Hodně drobných detailů, které přicházejí do hry při vaší normální interakci s různými programy (některých z nich si dosud nemusíte být vědomi). Nikdo nechce zadávat množství voleb každému programu, který se chystá spustit. Říkat mu, jaký druh terminálu se používá, hostitelské jméno počítače, jak by měl vypadat prompt...

Tak jako (So as coping) mechanismus, uživatelé mají cosi, co se nazývá prostředí. Prostředí definuje podmínky v nichž program běží a některé z těchto podmínek jsou proměnlivé. Uživatel je může měnit a hrát si s nimi, jak je to možné jedině v linuxovém systému. Téměř jakýkoliv shell bude mít proměnné prostředí (jestli ne, tak to asi není moc použitelný shell). Zde podáme přehled příkazů, které bash poskytuje pro manipulaci s jeho proměnnými prostředí.

```
$ set
```

set samotný vám ukáže vš echny proměnné prostředí, které jsou v současné chvíli definovány, a rovněž i jejich hodnoty. Tak jako většina vestavěných věcí v **bash**i, může dělat i pár dalších věcí (s parametry). Přenechme manuálovým stránkám **bash**(1), aby to vysvětlily pořádně. Dílčí ukázka výpisu z příkazu **set** na mém počítači vypadá takto:

```
PATH=/usr/local/lib/qt/bin:/usr/local/bin:/usr/bin:/bin:/usr/X11R6/bin:
 /usr/openwin/bin:/usr/games:.:/usr/local/ssh2/bin:/usr/local/ssh1/bin:
 /usr/share/texmf/bin:/usr/local/sbin:/usr/sbin:/home/logan/bin
 PIPESTATUS=([0]="0")
 PPID=4978
 PS1='\h:\w\$ '
 PS2='> '
 PS4='+ '
 PWD=/home/logan
 QTDIR=/usr/local/lib/qt
 REMOTEHOST=ninja.tdn
 SHELL=/bin/bash
```

Pověsíme si proměnné `PATH`, kterou jsme probírali dříve. Mohu spouštět cokoliv, co je v těchto adresářích pouhým zadáním jména souboru.

```
$ unset VARIABLE
```

unset odstraní všechny proměnné, které mu zadáte. Vymaže jak proměnné, tak i jejich hodnoty; **bash** zapomene, že vůbec kdy existovaly. (Don't worry. Unless it's something you explicitly defined in that shell session, it'll probably get redefined in any other session.)

```
$ export VARIABLE=some_value
```

Nyní, **export** je vskutku užitečný. Jeho použitím zadáte proměnné prostředí `VARIABLE` hodnotu "some_value". Pokud `VARIABLE` dosud neexistovala, od nynějška bude. Pokud `VARIABLE` už měla nějakou hodnotu, je přepsána. To není zrovna nejlepší, pokud se jen snažíte přidat další adresář do vaší `PATH`. V takovém případě asi budete chtít udělat tohle:

```
$ export PATH=$PATH:/some/new/directory
```

Povšimněte si použití `$PATH`: Když chcete, aby **bash** interpertoval proměnnou (tedy, aby jí nahradil její hodnotou), přidejte před jméno proměnné znak **$**. Například **echo $PATH** vypíše hodnotu proměnné `PATH`. V mém případě:

```
$ echo $PATH
 /usr/local/lib/qt/bin:/usr/local/bin:/usr/bin:/bin:/usr/X11R6/bin:
 /usr/openwin/bin:/usr/games:.:/usr/local/ssh2/bin:/usr/local/ssh1/bin:
 /usr/share/texmf/bin:/usr/local/sbin:/usr/sbin:/home/logan/bin
```

Kompletace pomocí tabulátoru

(Zde opět přichází něco moc bezva.)

1. Prostředí příkazové řádky znamená mnoho psaní.
2. Psaní je práce.
3. Nikdo nemá rád práci.

Z bodů 3 a 2 můžeme odvodit, že 4. nikdo nemá rád psaní. **bash** nás naštěstí uchrání před 5: Nikdo nemá rád prostředí příkazové řádky.

Ptáte se, jak **bash** může dokázat tento obdivuhodný čin? Kromě nahrazování žolíků popsaného dříve, **bash** ovládá kompletaci pomocí tabulátoru.

Dokončování tabulátorem pracuje takto: Píšete jméno souboru. Možná, že je ve vaší `PATH`, možná je vypisujete úplně. Jedniné, co teď musíte udělat je napsat dost znaků jména onoho souboru, aby byl jednoznačné identifikovatelný. Potom stiskněnte klávesu Tab. **bash** pochopí co chcete a dokončí psaní za vás!

Čas na příklad: Adresář `/usr/src` obsahuje dva podadresáře: `/usr/src/linux` a `/usr/src/sendmail`. Chci se podívat, co je v adresáři `/usr/src/linux`. Tak napíšu jenom **`ls /usr/src/l`**, stisknu klávesu **TAB** a **bash** mi dá `ls /usr/src/linux`.

Dále předpokládejme, že tu jsou dva adresáře `/usr/src/linux` a `/usr/src/linux-old`. Když napíšu **`/usr/src/l`** a stisknu **TAB**, **bash** vyplní tolik, kolik může a já dostanu `/usr/src/linux`. Tady se můžu zastavit, nebo můžu stisknout **TAB** ještě jednou a **bash** ukáže seznam adresářů, které vyhovují tomu, co jsem už napsal.

Proto méně psaní (a proto mohou mít lidé rádi prostředí příkazové řádky). Říkal jsem vám, že to je bezva.

Virtuální terminály

Třeba jste uprostřed nějaké práce a zjitíte, že byste potřebovali dělat něco jiného. Můžete prostě zahodit to, na čem jste pracovali a přepnout úlohy. Ale tohle je více−uživatelský systém, pravda? A vy se můžete přihlašovat simultánně kolikrát chcete, pravda? Tak proč byste měli dělat v jednu chvíli jen jednu věc?

Nemusíte. Sice nemůžeme mít vícero klávesnic, myší a monitorů u jednoho stroje; asi bychom to ani nechtěli. Jasně, hardware není tím řešením. Takže zbývá software − a v tuto chvíli nastupuje Linux: Poskytuje "virtuální terminály", neboli "VT".

Stisknutím Alt a funkční klávesy se můžete přepínat mezi virtuálními terminály. Každá funkční klávesa odpovídá jednomu z nich. Slackware může defaultně přihlašovat na 6 VT. Alt+F2 vás přenese na druhý VT, Alt+F3 na třetí, atd...

Ostatní funkční klávesy jsou rezervovány pro X sessions. Každá session X používá svůj vlastní VT, počínaje sedmým a následující. Když jste v X, pak je kombinace Alt+Funkční klávesa nahrazena Ctrl+Alt+Funkční klávesa. Takže jste−li v X a chcete se dostat zpátky do textového přihlašování (aniž byste ukončovali běh X), Ctrl+Alt+F3 vás přenese na třetí VT. A Alt+F7 vás přenese zpět do X.

Pozn.překl: K přepínání mezi VT v příkazové řádce můžete ještě používat Alt+šipka (vlevo, vpravo). Tj. o jeden terminál "dolů" nebo "nahoru". Např. jste−li v terminálu č.2, pak Alt+vlevo vás přenese na terminál č.1; Alt+vpravo vás přenese na terminál č.3.

Shrnutí

Tato kapitola pojednávala o uživatelích, shellu, příkazové řádce a vrituálních terminálech. Měli byste se cítit pohodlně při práci na příkazové řádce, při spouštění programů, používání rour, přesměrování a kombinování příkazů. A nakonec byste měli mít povědomí o síle superuživatele roota, a proč je špatná věc pracovat jako root pořád.

Práva

Práva (permissions) jsou další důležitou úlohou víceuživatelského aspektu souborového systému. S jejich pomocí můžete měnit kdo smí číst, zapisovat a spouštět soubory.

Informace o právech jsou ukládána jako čtyři osmičkové číslice, každá specifikuje jinou množinu práv. Jsou to práva pro vlastníka, skupinu a ostatní (world). Čtvrtá cifra je použitá k uložení speciálních informací jako je nastavené uživatelské ID, nastavené skupinové ID a "sticky" bit. Osmičkové hodnoty přidělené módům oprávnění jsou (mají též přidělená písmena, která zobrazují programy jako **ls** a mohou být využívána programem **chmod**):

Tabulka 9–1. Oktalové (osmičkové) hodnoty práv

Druh práva Permission Type	Osmičková hodnota	Písmenkové vyjádření
"sticky" bit	1	t
nastav ID uživatele	4	s
nastav ID skupiny	2	s
čtení	4	r
zápis	2	w
spouštění	1	x

Pro každou skupinu práv zadáváte osmičkové hodnotty. Například pokud chcete aby práva skupiny byla "čtení" a "zápis", měli byste pro práva skupiny zadat "6".

Defaultní práva v **bash**i jsou:

```
 $ ls -l /bin/bash
-rwxr-xr-x   1 root     bin  477692 Mar 21 19:57 /bin/bash
```

Pokud by šlo o adresář, bylo by na místě úvodní pomlčky "d". Dále jsou zobrazeny tři skupiny oprávnění (pro vlastníka, pro skupinu a pro ostatní). Vidíme, že vlastník má právo číst, zapisovat a spouštět (rwx). Skupina smí jenom číst a spouštět (r–x). A rovněž kdokoliv jiný smí pouze číst a spouštět (r–x).

Jak bychom nastavili práva k jinému souboru tak jak to dělá bash? Nejdříve si vytvořme soubor "example":

```
$ touch /tmp/example
$ ls -l /tmp/example
-rw-rw-r---  1 david    users    0 Apr 19 11:21 /tmp/example
```

Použijeme příkaz **chmod**(1) (což znamená "change mode" – "změň mód"), kterým nastavíme práva souboru "example". Nastavte osmičková čísla práv, která chcete. Aby vlastník mohl číst, psát a spouštět, měl by mít hodnotu 7. Číst a spouštět by bylo 5. Dejete to dohromady a zadejte to do příkazu **chmod**:

```
 $ chmod 755 /tmp/example
$ ls -l /tmp/example
-rwxr-xr-x   1 david    users    0 Apr 19 11:21 /tmp/example
```

K nastavení zvláštních oprávnění zadáme čísla do prvního sloupce. Například k nastavení ID uživatele a ID skupiny dáme do prvního sloupce "6":

```
 $ chmod 6755 /tmp/example
$ ls -l /tmp/example
-rwsr-sr-x   1 david    users    0 Apr 19 11:21 /tmp/example
```

Pokud vás osmičková čísla pletou, můžete místo nich používat písmena. Skupiny oprávnění pro **chmod** pak vypadají takto:

Vlastník	u
Skupina	g
Ostatní	o
Všichni zmínění	a

Abychom dosáhli toho, co nahoře, museli bychom použít několik příkazových řádků:

```
 $ chmod a+rx /tmp/example
$ chmod u+w /tmp/example
$ chmod ug+s /tmp/example
```

Někteří lidé dávají přednost písmenům před číslicemi. Obě cesty vyúsťují do té samé sady oprávnění.

Na několika místech jsme se zmínili o nastavování ID uživatele a ID skupiny. Možná si říkáte, o co jde? Obyčejně, když spustíte program, ten pak pracuje pod vaším uživatelským účtem. To znamená, že má všecha ta oprávnění, která máte vy jako uživatel. Totéž platí pro skupinu. Jestliže spustíte program, vykonává se pod vaší současnou skupinou. S nastaveným oprávněním ID uživatele si můžete vynutit, aby program vždy běžel jako majitel programu (třeba jako "root"). Nastavení ID skupiny je totéž, ale pro skupinu.

Zacházejte s tím opatrně. Nastavení ID uživatele a ID skupiny programům může ve vašem systému otevřít významné bezpečnostní mezery. Pokud často nastavujete ID uživatele programům, které vlastní "root", povolíte každému, aby provozoval tento program stejně jako root. A protože root nemá v systému žádná omezení, můžete si sami představit, jak velký bezpečnostní problém to je. Stručně řečeno, není špatné používat nastavení oprávnění ID uživatele a ID skupiny, ale dělejte to s citem.

Předchozí Struktura souborového systému
Výchozí Nahoru
Další Odkazy

Odkazy

Odkazy (links) jsou ukazovátka mezi soubory. Pomocí odkazů můžete mít soubory na mnoha místech a mít je přístupné pod mnoha názvy. Existují dva typy odkazů: Pevné a symbolické.

Pevné odkazy jsou názvy pro konkrétní soubor. Ty mohou existovat pouze uvnitř jediného adresáře a jsou vyjmuty jedině tehdy, když je ze systému vyjmuto reálné jméno. Jsou v určitých případech užitečné, ale mnoho uživatelů shledává symbolické linky mnohostrannější mi.

Symoblický odkaz, též nazývaný "měkký", může ukazovat na soubor vně svého adresáře. Je to vlastně malý souborek, obsahující informaci, kterou potřebuje. Symbolické odkazy můžete přidávat i mazat bez zasažení vlastního souboru

Odkazy nemají svou vlastní sadu práv či vlastnictví, ale místo toho je přebírají od souboru, na který ukazují. Slackware používá hlavně symbolické odkazy. Zde je obecný příklad:

```
  $ ls -l /bin/sh
lrwxrwxrwx   1 root     root     4 Apr  6 12:34 /bin/sh -> bash
```

Shell **sh** ve Slackwaru je vlastně **bash**. Odstraňování odkazů se dělá příkazem **rm**. K vytváření odkazů se používá příkaz **ln**. O těchto příkazech bude pojednáno podrobněji v 10. kapitole.

Pozn.překl: Tato kapitola mi připadá neuvěřitelně zmatená! Pro srovnání jsem tu nechal i původní anglický text. Jestli jsem ho jenom nepochopil, dejte mi vědět. grumpa@qwert.cz.:o)
Raději koukněte na manuálové nebo info stránky ve vašem systému (man ln, info ln).

Hard links are names for a particular file. They can only exist within a single directory and are only removed when the real name is removed from the system. These are useful in some cases, but many users find the soft link to be more versatile.

The soft link, also called a symbolic link, can point to a file outside of its directory. It is actually a small file containing the information it needs. You can add and remove soft links without affecting the actual file.

Links do not have their own set of permissions or ownerships, but instead reflect those of the file they point to. Slackware uses mostly soft links. Here is a common example:

The **sh** shell under Slackware is actually **bash**. Removing links is done using **rm**. The **ln** command is used to create links. These commands will be discussed in more depth in Chapter 10.

Linky – můj názor :o)

Jako překladatel jsem nechěl měnit původní text, takže až zde napíšu, jak toto téma – mimochodem velmi snadné – vímám já.

Každý soubor v Linuxu je ve skutečnosti označen číslem, kterému se říká nod. Aby to bylo srozumitelné i pro lidi, soubory mají i jména. Každý soubor má minimálně jedno jméno. Těm se říká "pevné" linky – odkazy. Na jeden soubor je tedy možné se odkazovat z různých adresářů pod různými jmény. Změní–li se práva k souboru u jednoho z názvů, projeví se ta změna i u všech ostatních. Smaže–li se pevný odkaz na soubor, nic se neděje s těmi ostatními odkazy. Je–li smazán i poslední pevný odkaz (tedy název) na soubor, pak je soubor z disku odstraněn. Příkazem ls s parametrem –l můžete zjistit, kolik linků na daný soubor ukazuje (druhý sloupeček ve výpisu).

Sybolické odkazy (soft links) jsou speciální souborky, které ukazují na nějaký pevný link, nebo adresář. Používají se hlavně jako "zkratky" k cestě z například z vašeho domovského adresáře do adresáře s dokumentací, ap. Symbolický odkaz má stejné vlastnosti a pracuje se s ním stejně jako s cílem na který odkazuje. Smaže–li se soubor, na nějž sybolický link ukazoval, pak symbolický link nezmizí, jen ztratí smysl života a propadne depresi. Smažte ho. Ať netrpí.

Pevný odkaz nemůže ukazovat na adresář, symbolický ano. Zamyslíte–li se nad tím, zjistíte, že to je logické a jinak to ani nejde.:o)

Odkazy se vytvářejí příkazem ln:

```
$> ln soubor [jmeno_odkazu]
```

Příkaz vytvoří odkaz na soubor. Není–li uveden druhý parametr, pak je odkaz vytvořen v aktuálním adresáři a dostane stejné jméno, jako originální soubor. Uvedeným způsobem vytvoříte odkaz pevný. Pro vytvoření symbolického odkazu přidejte za příkaz ln parametr –s.

```
$> ln -s soubor [jmeno_odkazu]
```

V tomto případě můžete na místě "soubor" zadat i jméno adresáře.

Připojování (mountování) zařízení

Jak jsme už pojednali v sekci *Prohlídka systému* ve 4 kapitole, všechny jednotky a zařízení ve vašem počítači jsou jedním velkým souborovým systémem. Různé oddíly na pevném disku, CD-ROM a diskety jsou umístěny v tom samém stromu. Abyste připojili tyto jednotky k souborovému sytému, abyste k nim mohli přistupovat, musíte použít příkazy **mount**(1) a **umount**(1).

Některá zařízení jsou připojována automaticky při startu systému. Tato zařízení jsou uvedena v souboru `/etc/fstab`. Chcete-li, aby něco bylo připojováno automaticky, uveďte příslušný příkaz do tohoto souboru. V ostatních případech budete muset zadávat příkaz k připojení a odpojení ručně, kdykoliv budete chtít zařízení použít.

fstab

Podívejme se na ukázku souboru `/etc/fstab`:

```
 /dev/sda1       /              ext2        defaults        1   1
/dev/sda2       /usr/local      ext2        defaults        1   1
/dev/sda4       /home           ext2        defaults        1   1
/dev/sdb1       swap            swap        defaults        0   0
/dev/sdb3       /export         ext2        defaults        1   1
none            /dev/pts        devpts      gid=5,mode=620  0   0
none            /proc           proc        defaults        0   0
/dev/fd0        /mnt            ext2        defaults        0   0
/dev/cdrom      /cdrom          iso9660     ro              0   0
```

V prvním sloupci je jméno zařízení. V našem příkladu těmito zařízeními jsou: Pět diskových oddílů rozkládajících se na dvou SCSI pevných discích, dále dva speciální souborové systémy, které nepotřebují zařízení, potom disketová jednotka a nakonec CD-ROM mechanika. Druhý sloupec uvádí adresář, pod nějž má být zařízení připojeno. Musí to být jméno adresáře – s výjimkou swapovacího oddílu. Třetí sloupec popisuje typ souborového systému na zařízení. Pro normální linuxový souborový systém to bude **`ext2`** (či nověji ext3). CD-ROM jednotky jsou iso9660 a zařízení s MS-Windows budou buď **`msdos`**, nebo **`vfat`**.

Čtvrtý sloupec obsahuje seznam voleb, které se uplatní na přpojovaný souborový systém. defaults vyhovuje téměř vždycky. Nicméně zařízení, která jsou read-only (jen ke čtení) by měla mít zadán příznak **`ro`**. Použitelných voleb je poměrně mnoho a my vás odkážeme na manuálové stránky `fstab`(5), kde je všechny najdete. Pátý sloupec využívá zálohovací utilita **dump** a šestý **fsck**. I tady vás odbydeme poukazem na manuálové stránky.

Při instalaci Slackware Linuxu sestaví většinu souboru **fstab** program "setup" sám. Vy jej budete potřebovat ručně editovat v případech, kdy například přidáváte nový disk, nebo chcete aby některá další zařízení byla připojována automaticky při startu systému.

mount a umount

Připojování dalších zařízení do systému je snadné. Jediné co musíte udělat, je použít příkaz **mount** – společně s několika volbami. Použití příkazu **mount** si můžete ještě zjednodušit, když si často připojované zařízení uvedete do souboru `/etc/fstab`. Řekněme například, že bych chtěl připojovat můj CD-ROM a že můj `/etc/fstab` vypadá tak, jak jsme si uvedli v předchozí sekci. Pak bych **mount** mohl použít takto:

```
# mount /cdrom
```

Jelikož pro tento přípojný bod mám příkaz v `/etc/fstab`, **mount** ví, jaké volby použít. Kdybch jej v `fstab` uvedený neměl, musel bych pro připojení zadat několik voleb:

```
# mount -t iso9660 -o ro /dev/cdrom /cdrom
```

I když příkazový řádek obsahuje tytéž informace, které byly v příkladu `fstab`, přesto si je ještě jednou projdeme. **`-t iso9660`** je typ souborového systému připojovaného zařízení. V tomto případě to je iso9660, který je na CD-ROM nejobvyklejší. **`-o ro`** říká, aby bylo zařízení připojeno jen pro čtení. **`/dev/cdrom`** je jméno připojovaného zařízení a **`/cdrom`** je umístění v souborovém systému, kam zařízení připojit.

Dříve než vyjmete disketu, CD-ROM či další vyjímatelné zařízení, které je právě připojené, musíte je odpojit. To se dělá příkazem **umount**. Neptejte se, kam zmizelo "n", protože to vám říct nedovedem. Jako argument příkazu **umount** můžete uvést buď název připojeného zařízení, nebo přípojný bod (mount point). Kdybyste tedy například chtěli odpojit CD-ROM připojený v předchozím příkladu, oba z následujících příkazů budou fungovat:

```
# umount /dev/cdrom
# umount /cdrom
```

Poznámka: Pokud vyjmete disketu, aniž byste ji **umount**-em odpojili, nebudou na ní nejspíš zapsaná data, která jste tam ukládali. Je to proto, že data se běžně ukládají do vyrovnávací paměti. Teprve zadání příkazu **umount** způsobí, že obsah vyrovnávací paměti se zapíše na disketu i fyzicky.

Připojování NFS

NFS je zkratka pro Network Filesystem (síťový souborový systém). Ten není fyzickou součástí vašeho souborového systému. Jen se pod něj připojuje.

Rozsáhlá Unixová síťová prostředí často sdílejí programy, domovské adresáře a mailový spool. Problém jak poskytnout identickou kopii těchto souborů všem počítačům je vyřešen používáním NFS. Ten můžeme použít například ke sdílení jedné skupiny domovských adresářů všemi pracovními stanicemi. Stanice, která si připojí adrsář přes NFS s ním pak může pracovat, jako by byl fyzicky uložen přímo na ní.

Další informace hledejte v sekci *NFS (Network File System)* v 5. kapitole a v manuálových stránkách `exports`(5), **nfsd**(8) a **mountd**(8).

Shrnutí

V této kapitole jste měli nabýt povědomí o vlastnictví a právech; měli byste vědět proč exitstují a jak je nastavit. Také byste měli vědět o odkazech mezi soubory, připojování zařízení a připojování pomocí NFS. Tyto tři věci jsou důležitými aspekty souborového systému. Měli byste mít základní představu, jak je používat.

cd

Příkaz **cd** se používá ke změně pracovního (aktuálního) adresáře. Jednoduše napíšete **cd** následovaný jménem cesty k adresáři. Zde je pár příkladů:

```
darkstar:~$ cd /bin
darkstar:/bin$ cd usr
bash: cd: usr: No such file or directory
darkstar:/bin$ cd /usr
darkstar:/usr$
```

Povšimněte si, že úvodní lomítko přestavuje cestu absolutní – od kořenového adresáře, zatímco bez úvodního lomítka zadáváte cestu relativní vzhledem k aktuálnímu adresáři. Proto druhý řádek skončil chybou – v adresáři /bin se podadresář "usr" nenachází, zatímco v kořenovém adresáři (4. a 5.řádek) ano.

Příkaz **cd** se od většiny ostatních příkazů trochu liší. Je to vestavěný příkaz shellu. O těchto příkazech jsme pojednali v sekci *Proměnné prostředí* v 8. kapitole. Možná vám to teď nic neříká, ale v základu to znamená, že pro **cd** neexistuje manuálová stránka. Místo "man" musíte použít shellovský help:

```
$ help cd
```

To vám vypíše možnosti příkazu **cd** a jak jej používat.

more

Příkaz **more** nazýváme "stránkovací utilita". Často se stává, že výstup nějakého příkazu je příliš dlouhý, aby se vešel na obrazovku. Jednotlivé příkazy neumějí zastavit svůj výpis, když je obrazovka plná. Přenechávají tuto práci stránkovací utilitě **more**.

Příkaz **more** zastaví po zaplnění obrazovky výpis a čeká až zmáčknete mezerník, aby mohl vypsat další porci řádků. Stiskem klávesy "enter" posunete výpis o řádek jediný. Zde je příklad:

```
$ cd /usr/bin
 $ ls -l
```

To bude chvíli rolovat... Abyste měli výpis zastavovaný po obrazovkách, prostě pošlete výpis rourou skzr more:

```
 $ ls -l | more
```

Tohle je označení pro rouru (pipe): "|". Rouru můžeme popsat jako "vezmi výstup z příkazu ls a předej ho jako vstup příkazu more". Rourou můžete příkazu **more** poslat v podstatě cokoliv, ne jenom výstup z **ls**. Práce s rourou je více popsána v sekci *Přesměrování vstupu a výstupu a roury* v 8. kapitole.

less

Příkaz **more** je fakt šikovný, ale často se vám stane, že přeběhnete obrazovku, na níž bylo to, co jste hledali. **More** nenabízí možnost vrátit se zpět. Tuto možnost ale nabízí obdobný příkaz **less**. Používá se všude tam, kde můžete použít příkaz **more**, takže předchozí příklady se vám na něj hodí taky. Takže **less** (anglicky "méně") je více než **more** (anglicky více).:o)

cat

cat je zkratka pro concatenate (spojovat). Původně byl tento příkaz zamýšlen pro spojování více textových souborů do jednoho, ale používá se i k dalším účelům.

Ke spojení dvou souborů do jednoho jednoduše napište jejich názvy za **cat** a výstup přesměrujte do souboru. Příkaz **cat** pracuje se standardním vstupem a standardním výstupem, takže musíte použít shellovské znaky pro přesměrování. Například:

```
$ cat file1 file2 file3 > bigfile
```

Tento příkaz vezme obsah souborů `file1`, `file2` a `file3` a sloučí je. Výstup je zapsán do souboru bigfile.

Můžete **cat** používat i k prostému zobrazní obsahu textového souboru na obrazovce monitoru.

```
$ cat soubor
```

Výpis samozřejmě můžete poslat rourou do **more** nebo **less**:

```
$ cat file1 | more
```

takže vám obsah souboru neprolítne monitorem, ale bude se ukazovat po částech.

Další rozšířené použití příkazu **cat** je pro kopírování souborů:

```
$ cat /bin/bash > ~/mybash
```

Program `/bin/bash` se takto zkopíruje do vašeho domovského adresáře a bude pojmenován "mybash".

Použití **cat** je mnohem rozsáhlejší, než to co jsme tu pojednali. Protože **cat** pracuje se stadardním vstupem i výstupem, je ideální pro použití v shellových skriptech jako součástka komplexnějších příkazů.

touch

(touch = dotknout se) Příkazu **touch** se používá jako časového razítka pro soubory. Tímto příkazem můžete nastavit čas posledního přístupu a čas změny. Pokud zadaný soubor neexistuje, **touch** jej vytvoří s nulovou velikostí (i to se ve speciálních případech používá). K označkování souboru aktuálním systémovým časem zadejte tento příkaz:

```
$ touch file1
```

Příkaz **touch** má i několik voleb, které určují například jaký časový údaj změnit, jaký čas použít a další. On–line manuálová stránka o tom pojednává podrobněji.

echo

Příkaz **echo** zobrazí zadaný text na obrazovce. Jeho použití je především v shellovských skriptech. Jednoduše zadáte žádaný řetězec za příkaz **echo**. Standardně **echo** za výpisem odřádkuje. To můžete potlačit pomocí volby –n. Volba –e způsobí, že **echo** bude v řetězci hledat escape znaky a vykonávat je.

```
$ echo To je dnes hezky!
To je dnes hezky!
```

Poznámka: Podle normy POSIX by se nověji místo příkazu echo měl používat příkaz **printf**.

mkdir

Příkazem **mkdir**(1) vytvoříte nový adresář. Následující příklad vytvoří adresář `manka` v aktuálním adresáři:

```
$ mkdir manka
```

Taky můžete zadat cestu:

```
$ mkdir /usr/local/manka
```

Volbou **-p** řekneme **mkidir**u, aby vytvořil i nadřízené adresáře. Výše uvedený příklad totiž zhavaruje, pokud `/usr/local` neexistuje. Volba **-p** zajistí i vytvoření `/usr/local`:

```
$ mkdir -p /usr/local/manka
```

ln

ln(1) se používá k vytvoření odkazu (link) mezi soubory. Tyto odkazy mohou být buď pevné (hard) nebo symbolické (soft). Rozdíly mezi nimi jsme popsali v sekci *Odkazy* v 9. kapitole. Kdybste chtěli vytvořit symbolický odkaz na adresář `/var/media/mp3` a umístit jej ve svém domovském adresáři, napište tohle:

```
$ ln -s /var/media/mp3 ~/mp3
```

Volba **`-s`** říká, aby byl vytvořen symbolický odkaz. Další parametr je cíl na který bude odkaz ukazovat. Poslední parametr je umístění a jméno odkazu. V tomto případě vznikne ve vašem domovském adresáři soubor `mp3`, který ukazuje na `/var/media/mp3`.

Vytvoření pevného odkazu je rovněž snadné. Jediné co uděláte je, že vynecháte volbu **`-s`**. Vytvoření pevného odkazu:

```
$ ln /var/media/mp3/song.mp3 ~/song
```

Už víme, že pevný odkaz nemůže ukazovat na adresář. Jen na soubor.

cp

Příkazem **cp**(1) se kopírují soubory. Uživatelé z DOSu si svými podobnosti s příkazem copy. Příkaz má mnoho zajímavých voleb a proto si prohlédněte jeho manuálovou stránku.

Nejobecnější využití tohoto příkazu je k prostému zkopírování souboru z jednoho místa do jiného. Například:

```
$ cp rumcajs /tmp
```

Takto zkopírujete soubor rumcajs z aktuálního adresáře do adresáře `/tmp`.

Při kopírování se změní některé parametry souboru jako například čas přístupu. Chcete-li zachovat co nejvěrněji vlastnosti originálu, použijte volbu -a:

```
$ cp -a rumcajs /tmp
```

Chcete-li rekurzívně kopírovat obsah adresáře do jiného adresáře, použijte tuto variantu příkazu:

```
$ cp -R adirectory /tmp
```

Takto se zkopíruje adresář `adirectory` do adresáře `/tmp`.

mv

Příkaz **mv**(1) přesouvá soubory z jednoho umístění do jiného. Uživatelé z DOSu vidí podobnost s příkazem move. Stačí zadat jméno souboru a kam se má přesunout, jak to vidíte v tomto příkladu:

```
# mv myfile /usr/local/share/hejaz
```

mv nabízí několik voleb, které jsou podrobně dokumentovány v manuálové stránce.

rm

(remove) Příkaz **rm**(1) maže soubory a adresářové stromy. Podobně jako del a deltree v DOSu. **rm** může být velmi nebezpečný, když na sebe nebudete dávat pozor. Narozdíl od DOSu či Windows, Linux nenabízí žádnou možnost jak smazaný soubor obnovit.

K vymazání jednoho souboru zadejte prostě jeho jméno za **rm**:

```
$ rm file1
```

Pokud by soubor neměl nastavená práva k zápisu, skončí takový pokus o vymazání chybou a neprovede se. Abyste si takové mazání vynutili, použijte volbu **-f**:

```
$ rm -f file1
```

K vymazání celého adresáře použijte společně volby **-r** a **-f**. Následuje ukázka toho, jak vymazat veškerý obsah pevného disku. Určitě to udělat nechcete, ale příkládek vám tu přesto uvedeme:

```
# rm -rf /
```

Buďte při používání **rm** velmi opatrní. Snadno se můžete střelit do vlastní nohy. Další volby pro tento příkaz jsou detailně probrány v manuálové stránce.

rmdir

Příkazem **rmdir**(1) se maž ou adresáře. Před vymazáním musí být adresář prázdný, jinak se vymazání neprovede. Syntaxe je jednoduchá:

```
$ rmdir <directory>
```

Následující příklad vymaž e podadresář `cipisek` z aktuálního adresáře:

```
$ rmdir cipisek
```

Kdyby adresář `cipisek` neexistoval, **rmdir** vám to poví.

Také můž ete zadat plnou cestu k vymazávanému adresáři, jak vidíme v tomto příkladě:

```
$ rmdir /tmp/karkulka
```

Tento příklad se pokusí vymazat adresář `karkulka` z adresáře `/tmp`.

Rovněž můž ete smazat adresář včetně jeho nadřízených adresářů zadáním volby **-p**:

```
$ rmdir -p /tmp/hejaz
```

Systém se nejdříve pokusí smazat adresář `hejaz` uvnitř adresáře `/tmp`. Pokud se to podaří, bude se dále snaž it smazat adresář `/tmp`. **rmdir** pokračuje v mazání tak dlouho, dokud se buď neobjeví chyba, nebo dokud není celý zadaný strom smazán.

Shrnutí

Tato kapitola ukázala množství programů, které manipulují soubory a adresáři. Měli byste vědět, jak vytvořit, smazat a přesunout v podstatě cokoliv, co se v souborovém systému nalézá. Taky byste měli vědět, jak soubory vypisovat a jak se jich dotknout (touch), je–li to třeba. A taky už víte, proč "**rm –rf /**" je dost blbej nápad.

Foregrounding

Pokud chcete komunikovat s upozaděným procesem, můžete jej přenést zpět do popředí. Máte–li upozaděný pouze jediný proces, můžete jej přenést do popředí tak, že napíšete:

```
$ fg
```

Jestliže program na pozadí dosud běžel, pak převezme kontrolu nad vaším terminálem a vy už nedostanete zpět prompt. Někdy se stane, že program, který běžel na pozadí také tam skončil svou práci. V takové případě byste obdrželi nějakou takovouhle hlášku:

```
  [1]+  Done              /bin/ls $LS_OPTIONS
```

To říká, že upozaděný proces (v tomto případě **ls** – to není teď důležité) skončil.

Je možné mít vícero upozaděných procesů najednou. Pokud se to stane, budete potřebovat vědět jak dostat do popředí jeden konkrétní proces. Pouhým zapsáním **fg** půjde dopředu proces, který byl upozaděn jako poslední. Ale co když máte na pozadí celou řadu procesů? Naštěstí bash obsahuje příkaz, umožňující všechny procesy vypsat. Ten příkaz se jmenuje **jobs** a poskytuje takovýto výstup:

```
$ jobs
 [1]   Stopped                 vim
 [2]-  Stopped                 amp
 [3]+  Stopped                 man ps
```

To vám ukazuje seznam všech procesů, které jsou na pozadí. Jak vidíte, všechy jsou pozastavené (stopped). Číslo na začátku je identifikátorem upozaděného procesu. Číslo za nímž je plus označuje proces, který by se dostal do popředí, kdybyste napsali pouze **fg**.

Pokud chcete dostat do popředí **vim**, musíte napsat:

```
$ fg 1
```

a **vim** se objeví zpět na konzoli. Backgrounding (upozaďování) procesů může být velmi užitečné, když máte pouze jeden terminál otevřený pro vytáčené připojení. Můžete mít několik programů běžících na tom jednom terminálu a průběžným přesouváním do popředí a do pozadí se mezi nimi přepínat.

ps

Takže teď už víte, jak přepínat do pozadí a zpět vícero procesů, které jste si spustili z příkazové řádky. A rovněž víte, že tu pořád běží celá řada procesů. Takže, jak si je vš echny zjistit? K tomu použijete příkaz **ps**(1). Tento příkaz má mnoho voleb. My tady popíš eme jen ty nejdůležitějš í z nich. Jejich úplný seznam najdete v manuálové stránce příkazu **ps**. Manuálové stránky jsou podrobně popsány v sekci *man* ve 2. kapitole.

Pouhým zapsáním **ps** získáte seznam programů, které běží na vaš em terminálu. Často to bývá dost stručný seznam:

```
$ ps
   PID TTY          TIME CMD
  7923 ttyp0    00:00:00 bash
  8059 ttyp0    00:00:00 ps
```

Ačkoliv se zdá, že to není mnoho procesů, ta informace je velmi typická. Dostanete ty samé sloupce bez ohledu na to, kolik proceů běží. Takže, co znamenají?

PID je ID procesu. Každý běžící proces má unikátní identifikátor. V jádrech 2.2.x toto číslo může být v rozmezí 1 až 32767. Každému procesu je přiděleno následující volné číslo PID. Když proces doběhne (nebo je zabit, jak uvidíte v následující sekci), uvolní svoje PID. Pokud by bylo dosaženo nejvyš šího možného čísla PID, budou se čísla přidělovat opět od nějniž š ích volných. U jader řady 2.4.x se tohle změnilo. Ta navíc používají 32−bitová PID.

Sloupec TTY říká, na kterém terminálu poces běží. Zadáním prostého ps získáme výpis pouze těch programů, které běží na aktuálním terminálu. Takže vš echy údaje v tomto sloupci budou stejné. Jak vidíte, oba uvedené procesy běží na ttyp0.

Sloupec TIME říká, kolik CPU času tento proces spotřeboval. To se liš í od skutečné doby, po kterou daný proces běží. Připomeňme si, že Linux je multitaskingový (víceúlohový) operační systém. V jednom okamžiku tu běží více pocesů najednou a každý tento proces si vezme malou část času procesoru. Takže sloupec TIME by měl pro každý proces ukazovat mnohem méně času, než jak dlouho už skutečně běží. Pokud tu uvidíte více než několik minut, může to znamenat, že něco není v pořádku.

Končně, sloupec CMD ukazuje, o jaký program vlastně jde. Tady se vypisuje jen základní jméno programu; ne už volby s nimiž byl spouš těn, či podobné informace. Abyste tyto informace získali, budete muset použít jednu z mnoha voleb příkazu **ps**. Stručně si je probereme.

Můžete získat úplný seznam procesů, běžících na vaš em systému, když použijete správnou kombinaci voleb. To pravděpodobně vyústí v předlouhý seznam procesů (padesát pět na mám laptopu, právě když píš u tuhle větu). Takže já ten výstup zkrátím:

```
$ ps -ax
 PID TTY      STAT   TIME COMMAND
   1 ?        S      0:03 init [3]
   2 ?        SW     0:13 [kflushd]
   3 ?        SW     0:14 [kupdate]
   4 ?        SW     0:00 [kpiod]
   5 ?        SW     0:17 [kswapd]
  11 ?        S      0:00 /sbin/kerneld
  30 ?        SW     0:01 [cardmgr]
  50 ?        S      0:00 /sbin/rpc.portmap
  54 ?        S      0:00 /usr/sbin/syslogd
  57 ?        S      0:00 /usr/sbin/klogd -c 3
  59 ?        S      0:00 /usr/sbin/inetd
  61 ?        S      0:04 /usr/local/sbin/sshd
  63 ?        S      0:00 /usr/sbin/rpc.mountd
  65 ?        S      0:00 /usr/sbin/rpc.nfsd
  67 ?        S      0:00 /usr/sbin/crond -l10
  69 ?        S      0:00 /usr/sbin/atd -b 15 -l 1
  77 ?        S      0:00 /usr/sbin/apmd
  79 ?        S      0:01 gpm -m /dev/mouse -t ps2
  94 ?        S      0:00 /usr/sbin/automount /auto file /etc/auto.misc
 106 tty1     S      0:08 -bash
 108 tty3     SW     0:00 [agetty]
 109 tty4     SW     0:00 [agetty]
 110 tty5     SW     0:00 [agetty]
 111 tty6     SW     0:00 [agetty]
 [výstup zkrácen]
```

Větš ina těch procesů je na větš ině systémů spouš těna v čase bootování. Já jsem ve svém systému provedl několik úprav, takže váš výpis se bude pravděpodobně dost liš it. Nicméně řadu těch procesů uvidíte na vaš em systému taky. Jak vidíte, pomocí −ax se zobrazily u jmen procesů i volby, s nimiž byly ony procesy spouš těny. Také to přidalo další sloupec a nějaký zajímavý výstup.

První čeho si vš imnete je, že větš ina těch procesů má uvedeno, že běží na tty ?. To jsou procesy, které byly spouš těny z terminálu, který už není aktivní. Proto už nejsou spojeny s žádným konktrétním terminálem

Dále je tu nový sloupec STAT. Ten ukazuj stav (status) procesu. S znamená "spící": Proces čeká na něco, co se má stát. Z znamená "zombie" (nemrtvý) proces. To je takový proces, jehož rodič zemřel a zanechal po sobě proces−děcko (child process). To není dobrá věc.

Chcete−li vidět ješ tě více informací o běžících procesech, vyzkouš ejte tohle:

```
$ ps -aux
 USER       PID %CPU %MEM   VSZ  RSS TTY      STAT START   TIME COMMAND
 root         1  0.0  0.0   344   80 ?        S    Mar02   0:03 init [3]
 root         2  0.0  0.0     0    0 ?        SW   Mar02   0:13 [kflushd]
 root         3  0.0  0.0     0    0 ?        SW   Mar02   0:14 [kupdate]
 root         4  0.0  0.0     0    0 ?        SW   Mar02   0:00 [kpiod]
 root         5  0.0  0.0     0    0 ?        SW   Mar02   0:17 [kswapd]
 root        11  0.0  0.0  1044   44 ?        S    Mar02   0:00 /sbin/kerneld
 root        30  0.0  0.0  1160    0 ?        SW   Mar02   0:01 [cardmgr]
 bin         50  0.0  0.0  1076  120 ?        S    Mar02   0:00 /sbin/rpc.port
 root        54  0.0  0.1  1360  192 ?        S    Mar02   0:00 /usr/sbin/sysl
```

```
root        57  0.0  0.1  1276  152 ?        S    Mar02   0:00 /usr/sbin/klog
root        59  0.0  0.0  1332   60 ?        S    Mar02   0:00 /usr/sbin/inet
root        61  0.0  0.2  1540  312 ?        S    Mar02   0:04 /usr/local/sbi
root        63  0.0  0.0  1796   72 ?        S    Mar02   0:00 /usr/sbin/rpc.
root        65  0.0  0.0  1812   68 ?        S    Mar02   0:00 /usr/sbin/rpc.
root        67  0.0  0.2  1172  260 ?        S    Mar02   0:00 /usr/sbin/cron
root        77  0.0  0.2  1048  316 ?        S    Mar02   0:00 /usr/sbin/apmd
root        79  0.0  0.1  1100  152 ?        S    Mar02   0:01 gpm
root        94  0.0  0.2  1396  280 ?        S    Mar02   0:00 /usr/sbin/auto
chris      106  0.0  0.5  1820  680 tty1     S    Mar02   0:08 -bash
root       108  0.0  0.0  1048    0 tty3     SW   Mar02   0:00 [agetty]
root       109  0.0  0.0  1048    0 tty4     SW   Mar02   0:00 [agetty]
root       110  0.0  0.0  1048    0 tty5     SW   Mar02   0:00 [agetty]
root       111  0.0  0.0  1048    0 tty6     SW   Mar02   0:00 [agetty]
[výstup zkrácen]
```

To už je docela dost informací. Máme tu nově informaci o tom, který už ivatel proces spustil, kolik systémových zdrojů proces využ ívá (%CPU, %MEM, VSZ a RSS), a kdy byl proces spuš těn. To už je obvykle dostatek informací, které mohou být systémovému administrátorovi k už itku. To zase přináš í dalš í potíž : Informace teď vybíhají za pravý okraj obrazovky, takž e je nevidíte vš echny. Spravíme to použ itím volby –w.

Mož ná ten výstup není moc hezký, ale dělá svou práci. Nyní máte úplný výpis pro kaž dý proces. Jsou ješ tě dalš í informace, které si o jednotlivých procesech můž ete nechat vypsat. Počtěte si o tom ve velmi podrobné manuálové stránce pro **ps**. Nicméně volby, které jsme si ukázali jsou nejoblíbenějš í a stanou se asi těmi, které budete potřebovat nejčastěji.

Předchozí
Foregrounding

Výchozí
Nahoru

Dalš í
kill

kill

Občas se stane, že se nějaký proces začne divně chovat, a vy jej potřebujete odstranit. Program, sloužící k takovéto administraci se nazývá **kill**(1) (zabít) a k ovládání procesů může být použit několika způsoby. Nejobvyklejším použitím příkazu **kill** je zabití procesu. Budete to potřebovat, když se nějaký proces zaběhne a používá příliš mnoho systémových zdrojů, nebo když vás nějaký proces prostě jenom štve.

Abyste mohli proces zabít, budete potřebovat znát buď jeho PID, nebo jeho jméno. Pro získání PID použijte příkaz **ps**, který jsme probrali v předchozí sekci. Například, abyste zabili proces s číslem PID 4747, napište tohle:

```
$ kill 4747
```

Upozorňuji, že musíte být vlastníkem procesu, abyste ho mohli zabít. To je bezpečnostní opatření. Kdybyste měli právo zabíjet procesy spuštěné ostatními uživateli, asi by to způsobilo pár zmatků. Nicméně root může zabít jakýkoliv proces v systému.

Ještě máme další variantu příkazu kill, nazvanou **killall**(1). Tento program dělá přesně to, co říká (zabij všechny): Zabíjí všechny běžící procesy zadaného jména. Kdybyste chtěli zabít všechny běžící procesy "vim", napsali byste následující příkaz:

```
$ killall vim
```

A všechny procesy **vim**, které máte spuštěné, zemřou. Uděláte-li to jako root, zabijete všechny **vim** spuštěné všemi uživateli. Následující příklad ukazuje, jak vykopnout všechny uživatele (včetně sebe) ze systému:

```
# killall bash
```

Někdy pouhý **kill** na svou práci nestačí. Některé procesy pomocí **kill** prostě nezemřou. Na takové budete potřebovat účinnější formu. Pokud třeba takové urputné PID 4747 neodpovědělo na váš vražedný požadavek, měli byste udělat následující:

```
$ kill -9 4747
```

To téměř jistě přivede proces 4747 k smrti. Totéž můžete udělat pomocí **killall**. Co se přihodilo je to, že byl procesu zaslán jiný signál. Obyčejný kill posílá procesům signál SIGTERM (terminate – ukončit). **kill -9** posílá procesům signál SIGKILL (kill – zabít). K dispozici máte celou řadu různých signálů. Jejich seznam můžete získat napsáním:

```
$ kill -l
   1) SIGHUP     2) SIGINT    3) SIGQUIT   4) SIGILL
   5) SIGTRAP    6) SIGABRT   7) SIGBUS    8) SIGFPE
   9) SIGKILL   10) SIGUSR1  11) SIGSEGV  12) SIGUSR2
  13) SIGPIPE   14) SIGALRM  15) SIGTERM  17) SIGCHLD
  18) SIGCONT   19) SIGSTOP  20) SIGTSTP  21) SIGTTIN
  22) SIGTTOU   23) SIGURG   24) SIGXCPU  25) SIGXFSZ
  26) SIGVTALRM 27) SIGPROF  28) SIGWINCH 29) SIGIO
  30) SIGPWR
```

Číslo se používá společně s příkazem **kill**, zatímco jméno bez úvodního SIG se používá s příkazem **killall**. Tady je další příklad:

```
$ killall -KILL vim
```

Poslední možností jak využít příkaz **kill** je použít ho k restartování nějakého procesu. Zasláním signálu SIGHUP přimějete většinu procesů, aby znovunačetly své konfigurační soubory. To je obzvláště užitečné, potřebujete-li nějaký systémový proces přimět k znovunačtení konfiguračního souboru poté, co jste jej upravili.

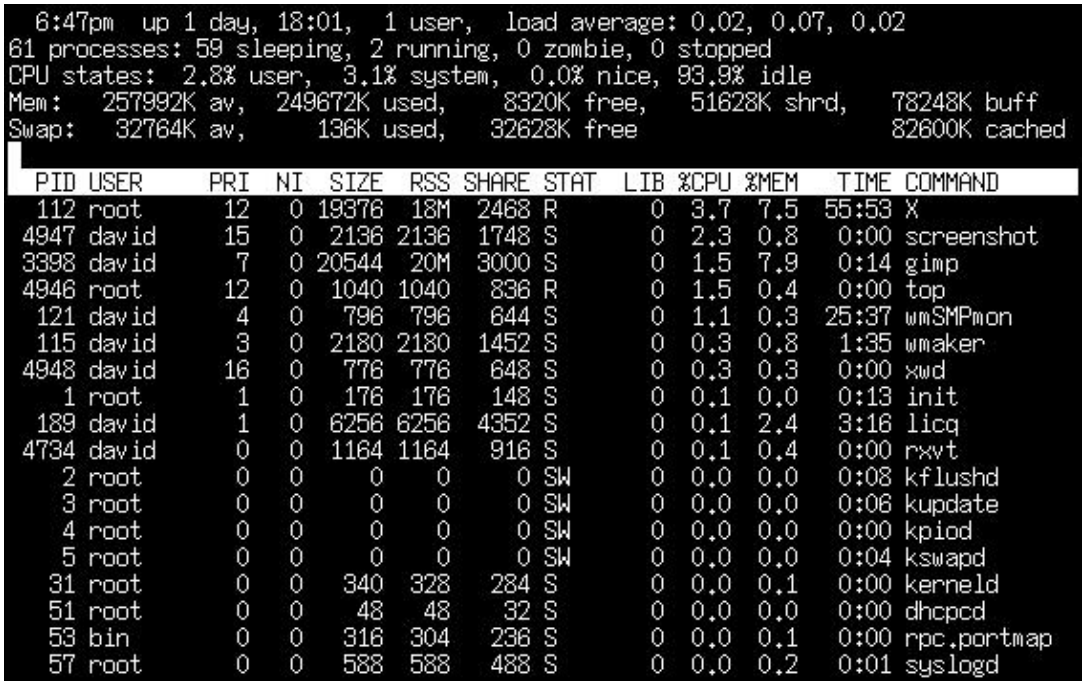
top

Na závěr tu máme příkaz, který můžete využít ke zobrazení obnovující se informace o procesech, které v systému běží. Ten příkaz se jmenuje **top**(1) a spouští se takto:

```
$ top
```

Tohle zobrazí plnou obrazovku informací o procesech běžících v systému. Také to vypíše některé obecné informace o systému samotném: Průměrnou zátěž (load average), počet procesů, stav CPU, informaci o volné paměti. Podrobnosti o procesech zahrnují PID, jméno uživatele, prioritu, využití CPU a paměti procesem, jak dlouho proces běží a jaké je jeho jméno.

Figure 11–1. Example output of the `top` program.



Ten příkaz se jmenuje **top** (vršek), protože na vrcholu výpisu jsou uvedeny procesy, které nejvíce zatěžují systém. Pikantní je, že právě samotný **top** bude vypsán jako úplně první (na většině systémů:o) – z důvodu využití CPU. Nicméně **top** je celkem užitečný ke zjištění, který proces se chová podivně a měl by být zabit.

Shrnutí

Tato kapitola pojednávala o tom co to je proces a jak můžeme procesy ovládat. To zahrnuje backgrounding a foregrounding a příkazy **ps**, **top** a **kill**, abyste je udrželi na uzdě. Měli byste být schopni zjistit které procesy běží na vašem systému a jak se jich zbavit, když se přestanou chovat správně.

Správné vypínání systému

Je velmi důležité, aby byl systém vypínán správně. Prosté vypnutí vypínačem může způsobit vážná poškození v souborovém systému. Pokud je systém v činnosti, jsou soubory používány, a to i když vy neděláte nic. Vzpomeňte si, že množství procesů běží po celý čas na pozadí. Tyto procesy spravují systém a udržují si několik souborů otevřených. Je–li systém prostě vypnut, tyto soubory nebudou správně uzavřeny a poškodí se. V závislosti na tom, které soubory byly poškozeny může být poškozen i celý systém trvale. V každém případě vás čeká nutnost projít dlouhou procedurou kontroly souborového systému při příštím nastartování počítače.

Když se tedy chystáte rebootovat, nebo vypnout váš počítač, je velmi důležité udělat to správně. Je několik způsobů jak to udělat: Můžete si vybrat kterýkoliv se vám zdá nejzábavnější. Většina metod pro ukončování systému může být použita rovněž pro rebootování.

První metodou je použití programu **shutdown**(8), a je také asi nejoblíbenější. Program **shutdown** může být používán k ukončování i k rebootování ve stanovený čas a může zobrazovat zprávy ostatním uživatelům přihlášeným do vašeho počítače o tom, že systém se chystá ukončit svou činnost.

Základní způsob použití programu **shutdown** pro zastavení počítače je:

```
# shutdown -h now
```

V tomto případě jsme neposílali uživatelům žádnou speciální zprávu, takže se jim zobrazí defaultní zpráva programu **shutdown**. Parametr "now" (teď) určuje čas, kdy se má příkaz provést a "–h" znamená "halt" (zastavit) systém. No, není to právě nejpřátelštější způsob zastavení víceuživatelského systému, ale vašem domácím počítači to bude vyhovující. Na víceuživatelském systému bychom měli měli dát ostatním uživatelům trochu času:

```
# shutdown -h +60
```

Tohle by zastavilo systém až za hodinu (60 minut). To je obyčejně dostatečná doba na normálním víceuživatelském systému. Skutečně významné systémy by měli svůj čas pro ukončování práce plánovat pokročilejším způsobem a zasílat informaci o tomto záměru pomocí `/etc/motd`(5) (soubor obsahující zprávu, která se zobrazí každému uživateli bezprostředně poté, co se přihlásí do systému – pozn.překl).

Pro rebootování systému používáme stejný příkaz, pouze místo "–h" napíšete "–r":

```
# shutdown -r now
```

S parametrem "–r" můžete samozřejmě použít stejný způsob zápisu času, který jsme ukázali u parametru "–h". Mnoho dalších věcí můžet dělat s příkazem **shutdown**. Na podrobnosti se podívejte do manuálových stránek.

Druhý způsob jak ukončit práci s počítačem je použít příkazy **halt**(8) a **reboot**(8). Jak jejich názvy naznačují, **halt** bezprostředně začne s ukončováním činnosti počítače a **reboot** rebootuje systém. Reboot je jednoduše symbolickým linkem na halt. Voláme je takto:

```
# halt
# reboot
```

Nízko–úrovňová metoda rebootování či zastavení systému je promluvit přímo k procesu **init**. Všechny ostatní metody jsou jen příjemnějším způsobem, jak hovořit k **init**u. Ale můžete k němu hovořit přímo, používáte–li příkaz **telinit**(8) (všimněte si, že je tam jen jedno "l"). Jeho pomocí řeknete initu, do jakého runlevelu se má přepnout. To způsobí spuštění speciálního skriptu. Tento skript zabije nebo (spawn) procesy, jak je pro daný runlevel potřeba. Funguje to i pro rebootování a ukončování, protože obě tyto situace jsou rovněž – speciálními – runlevely.

```
# telinit 0
```

Runlevel 0 je halt mód. Sdělením initu, aby vstoupil do runlevelu 0 způsobí, že všechny procesy budou zabity, souborové systémy odmountovány a počítač zastaven. To je zcela přijatelný způsob jak vypnout systém. Na mnoha laptopech to také ve finále způsobí i vypnutí počítače (u "stolních" počítačů s novějšími motherboardy můžete zkusit příkaz "poweroff", který vypne počítač taky – pozn.překl.).

```
# telinit 6
```

Runlevel 6 je rebootovacím módem. Všechny procesy budou zabity, souborové systémy odmountovány a počítač rebootován. To je zcela přijatelný způsob jak systém rebootovat.

A máme tu ještě jednu metodu pro rebootování systému. Všechny dosud uvedené metody vyžadovaly, abyste byli rootem. Nicméně je možné rebootovat počítač, i když rootem nejste. Pouze to vyžaduje fyzický přístup ke klávesnici. Vykonáním "tříprstového pozdravu" (control–alt–delete) zahájíte okamžitý reboot počítače. Tímto způsobem ve skutečnosti spouštíte program **/usr/sbin/ctrlaltdel**(8). Je tedy možné, že vám to fungovat nebude má–li tento program neobvyklá práva ke spouštění, nebo dokonce v systému chybí.

Shrnutí

Tato kapitola pojednávala o procedurách pro přidání a vyjmutí uživatelů a skupin. Měli byste umět provádět tyto úkoly za použití dodávaných skriptů a nebo ručně. K tomu byste měli i vědět, co se při přidávání uživatele děje. Měli byste znát techniky pro volbu hesla a jak měnit uživatelské informace. Konečně byste měli vědět, jak ohleduplně vypínat systém a proč je to tak důležité. Toto jsou všechno důležité součásti administrace systému, bez ohledu na to, je–li to váš domácí počítač, nebo velký síťový server.

finger

finger(1) vrací informace o zadaném uživateli. Zadáte programu **finger** nějaké jméno uživatele či nějakou e−mailovou adresu a on se pokusí kontaktovat příslušný server a zjistit uživatelské jméno, kancelář, telefonní číslo a další informace. Zde je příklad:

```
$ finger johnc@idsoftware.com
```

finger vrací uživatelské jméno, stav e−mailu, telefonní čísla a obsah souborů označovaných jako "dot plan" a "dot project". Zasílané informace se mohou lišit podle toho, jaký finger server na daném stroji běží. Ten, který se dodává se Slackwarem vrací defaultně následující informace:

- uživatelské jméno
- plné jméno
- domovský adresář
- používaný shell
- číslo místnosti
- telefon do práce
- telefon domů
- kdy byl naposledy (nebo jak dlouho právě je) připojen
- stav e−mailu
- obsah souboru `.plan`, který se nachází v jeho domovské adresáři
- obsah souboru `.project`, který se nachází v jeho domovské adresáři

Údaje o plném jméně, místnosti a telefonních číslech se nastavují pomocí příkazu **chfn**. Ten tyto údaje uloží do souboru `/etc/passwd`. Ke změně souborů .plan a .project použijte jednoduše váš oblíbený textový editor. Tyto soubory musí být umístěny ve vašem domovském adresáři.

Mnoho lidí "fingeruje" svůj vlastní účet ze vzdáleného počítače proto, aby rychle zjistili, nemají−li nový e−mail. Rovněž můžete vidět aktuální plány či projekt daného uživatele. Takový John Carmack z id Software svůj .plan soubor pravidelně aktualizuje, aby sděloval uživatelské komunitě, na čem právě pracuje.

Jako jiné příkazy i **finger** má možnost zadávat volby. Na ně se už podívejte do manuálové stránky.

telnet

Kdysi kdosi prohásil, že **telnet**(1) je ta neskvělejší věc, jaká kdy byla na počítačích k vidění. Možnost vzdáleně se přihlásit a pracovat na jiném počítači je to, co odlišuje Unix a Unix−like operační systémy od ostatních OS.

telnet vám umožňuje přihlásit se do vzdáleného počítače, jako byste seděli u jeho terminálu. Jakmile je ověřeno vaše uživatelské jméno a heslo, dostanete shell prompt a můžete dělat cokoliv, co na textové konzoli dělat jde. Tvořit e−maily, číst newsové skupiny, přesouvat soubory dokolečka, a tak dále. Pokud vám běží X a při−telnetujete do vzdáleného počítače z xterm−inálu, můžete na vzdáleném systému spouštět i X−kové programy a zobrazené je mít na svém. Pokud vás tohle zajímá, hledejte podrobnosti v 6. kapitole v sekci *Exporting displays* (snad i to někdy přeloží:o))

K přihlášení do vzdáleného stroje použije následující syntaxi:

```
$ telnet <hostname>
```

Pokud host odpovídá, obržíte výzvu k přihlášení. Zadejte vaše uživatelské jméno a heslo. A je to. Jste v shellu. K ukončení vaší **telnet**−ové session použije buď příkaz **exit**, nebo příkaz **logout**.

> DŮLEŽITÉ UPOZORNĚNÍ: **telnet** nekóduje (nešifruje) informace, které posílá. Vše je zasíláno ve formě čistého textu; včetně hesel. Proto se nedoporučuje používat **telnet** k připojování přes Internet. Místo něj používejte Secure Shell (SSH). Ten kóduje veškerý provoz a je volně a zdarma k dispozici. Více informací získáte na stránkách http://www.ssh.org/
> (pozn.překl: V současnosti je ssh normální součástí distribuce Slackwaru. Používá se v podstatě stejně.)

FTP klienti

FTP je zkratka od File Transfer Protocol (protokol pro přenos souborů). Ten vám umožňuje zasílat a přijímat soubory mezi dvěma počítači. Máme FTP server a FTP klienta. V této sekci pohovoříme o klientovi.

Pro zvědavé: "klient" jste vy. "Server" je počítač, který odpovídá na vaše FTP požadavky a umožňuje vám přihlásit se. Soubory budete stahovat z a odesílat na server. Klient neumí přijímat požadavky na FTP spojení. Pouze o to umí sám žádat servery.

ftp

Pro připojení na nějaký FTP server jednoduše spusťte program **ftp**(1) a zadejte jméno hostitele:

```
$ ftp <hostname>
```

Pokud na hostiteli běží FTP server, bude se vás ptát na uživatelské jméno a heslo. Můžete se přihlásit jako vy, nebo jako "anonymous". Anonymní FTP sajty jsou velice oblíbené k udržování archivů softwaru. Například abyste získali Slackware Linux prostřednictvím FTP, musíte použít anonymní přihlášení.
(Stejně jako v minulé kapitole i tady zapoměl autor zmínit, že při přihlašování jménem a heslem musí tuto kombinaci jméno/heslo vzdálený hostitel znát. Jinými slovy, musí vám tam zřídit váš účet – pozn.překl.).

Jakmile jste připojeni, objeví se vám na monitoru `ftp>` prompt. FTP používá vlastní speciální příkazy, které jsou ovšem často podobné standardním shellovým příkazům. Následující tabulka ukazuje pár základních příkazů a vysvětluje co dělají:

Tabulka 13–1. ftp příkazy

Příkaz	Účel
ls	Vypíše názvy souborů v aktuálním adresáři vzdáleného počítače
cd <adresar>	Změna adresáře
get <jmeno_souboru>	Stáhne soubor
put <jmeno_souboru>	Odešle soubor
hash	Přepíná (ne)vypisování "#" indikátoru za každý přenesený buffer
prom	Přepíná interaktivní režim pro stahování
mget <maska>	Stáhne jeden nebo více souborů. Při zadávání jmen můžete používat "wildcards" (* a ?)
mput <maska>	Odešle jeden nebo více souborů. Při zadávání jmen můžete používat "wildcards" (* a ?)
quit	Odhlášení od FTP serveru

Používání programu FTP je dost jednoduché, ale postrádá uživatelské rozhraní, na jaké je v současnosti větší ina z nás zvyklá. Manuálová stránka vysvětluje další volby, i pro příkazovou řádku, které lze **ftp**(1) zadat.

ncftp

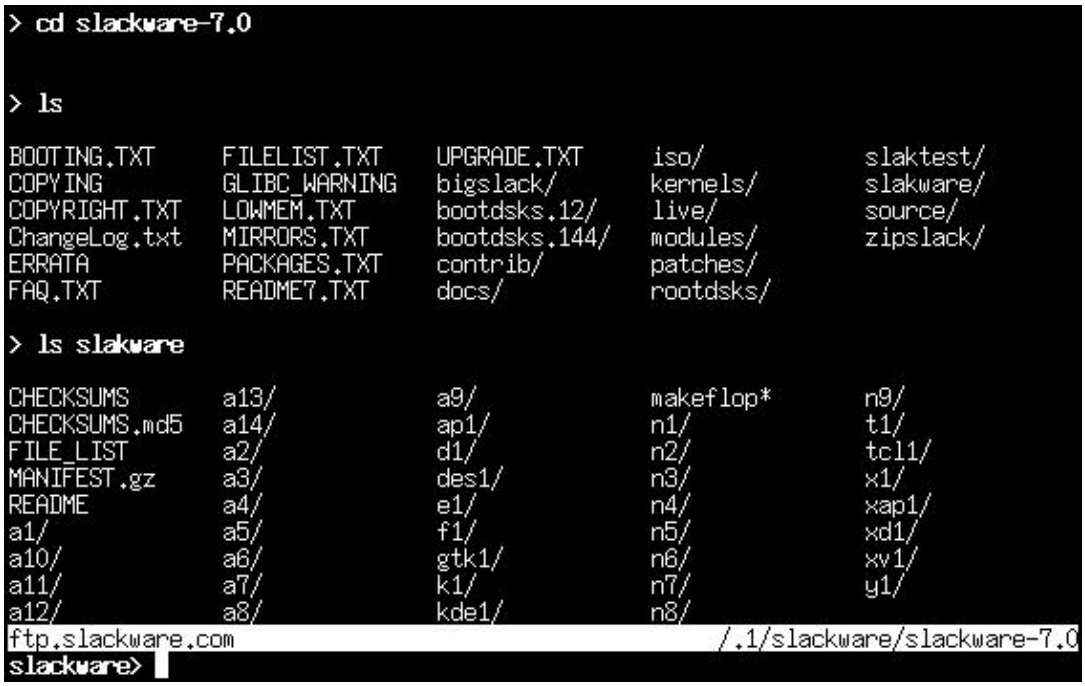
ncftp(1) (vyslovováno jako "Nik–F–T–P") je alternativou k tradičnímu **ftp** klientovi dodávanému se Slackwarem. Je to stále ještě textově–orientovaný program, ale nabízí mnoho výhod před **ftp**, včetně:

- dokončování tabulátorem*
- Bookmarks – záložky
- Pasivní a ne–pasivní módy FTP přenosu*
- Liberálnější používání wildcard*
- Historii příkazů*

* – (pozn.překl: Vše se vyvíjí, takže tradiční ftp tyto vymoženosti už také zná. Ale vyvíjí se i ncftp, takže stále může vést:o)

Defalutně bude **ncftp** zkoušet přihlásit se ke vzdálenému serveru anonymně. Můžete si vyžádat přihlašovací výzvu zadáním volby "–u". Jakmile jste přihlášeni, můžete používat tytéž příkazy, jako v **ftp**.

Obrázek 13–1. Příklad obrazovky s NcFTP.



Prev
telnet

Home
Up

Next
email

email

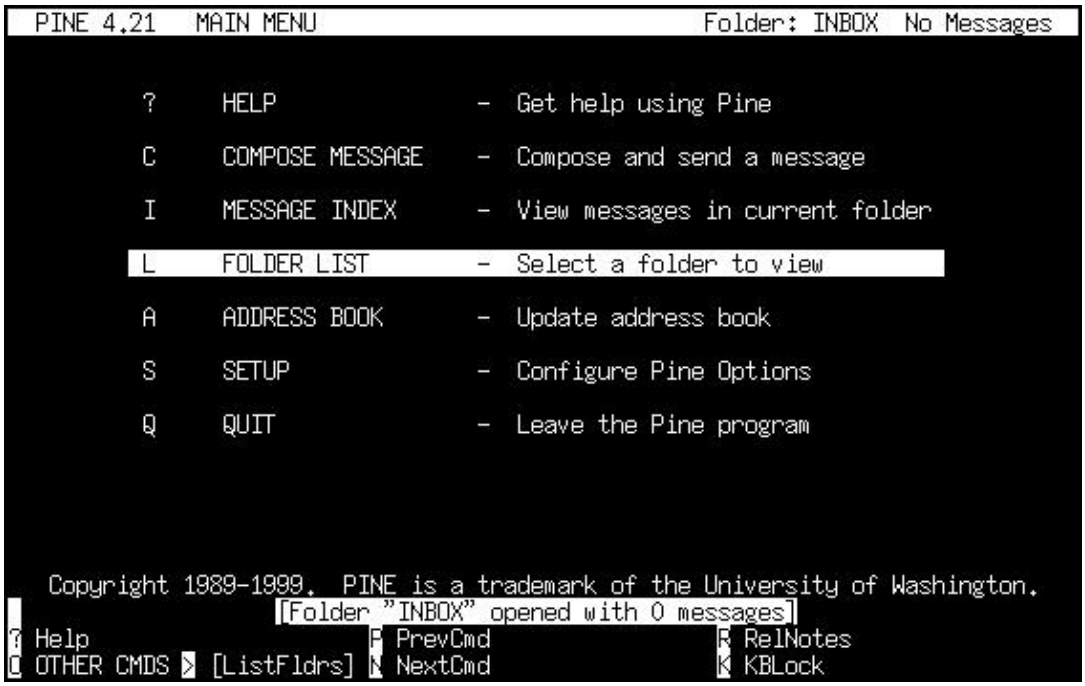
Elektronická pošta je jedna z nejoblíbenějších věcí, které můžeme na Internetu dělat. V roce 1998 se psalo, že bylo posláno více e-mailů než normální pošty. E-mail je skutečně rozšířený a užitečný.

Ve Slackwaru nabízíme standardní mailový server a několik mailových klientů. Všechny klientské mailové programy o nichž tu pojednáme budou textově orientované. Mnoho uživatelů Windows možná bude protestovat, ale uvidíte, že textové orientovaný mailový klient je velice výhodný, zvláště chcete-li poštu kontrolovat na dálku.

pine

pine(1) není **elm**. Či jak zní to rčení. Washingtonská univerzita vytvořila tento svůj snadno ovladatelný program pro Internteové news a e-mail pro potřeby svých studentů. **pine** je jedním z nejoblíbenějších e-mailových klientů, které se dnes používají, a to pro svou příchuť jak Unixovou, tak Windowsovou.

Obrázek 13-2. Hlavní menu programu Pine.



Vidíte nabídku příkazů a řádek příkazových klávesových zkratek dole. **pine** je vskutku komplexnim programem, a proto se tu nemůžeme věnovat všem jeho vlastnostem.

Abyste viděli, co máte v inboxu (došlé poště), stiskněte **i**. Vypíše se seznam zpráv, kde vidíte datum, autora a předmět. Najeďte si na zprávu, kterou chcete číst a stiskněte **enter**. Chcete-li na ni odpovědět, stikněte **r**. Jakmile máte odpověď hotovou, odešlete jí stiskem **Ctrl+X**. Můžete stisknout **i** a dostanete se zpět na seznam zpráv.

Chcete-li nějakou zprávu smazat, stiskněte **d**. To označí zprávu, jako že má být smazána. **pine** maže maily až když opouštíte program. **pine** rovněž umožňuje uchovávat zprávy ve složkách. Seznam existujících složek získáte, stisknete-li **l**. Chcete-li zprávu uložit do jiné složky, stiskněte **s**. Pine se vás pak zeptá na jméno složky, kam má zprávu uložit.

pine nabízí mnoho, mnoho možností; rozhodně byste se měli podívat do manuálové stránky tohoto programu. Najdete tam nejčerstvější informace o něm.

elm

elm(1) je dalším oblíbeným textové orientovaným e-mailovým klientem. Sice není až tak uživatelsky přítulný jako **pine**, ale je tu zas o něco déle.

Obrázek 13-3. Hlavní obrazovka programu Elm.

Vaší výchozí pozicí v programu elm je váš inbox. U zpráv je uvedeno číslo, datum, odesílatel a předmět. K výběru zprávy použijte klávesy se šipkami a pak stiskněte Enter.

Chcete-li psát novou zprávu, pak na hlavní obrazovce stiskněte **m**. Klávesou **d** označíte zprávu k vymazání. Klávesou **r** se dostanete k psaní odpovědi na aktuálně zobrazenou zprávu. Všechny tyto klávesy jsou zobrazeny v dolní části obrazovky nad promptem.

Manuálová stránka pojednává o programu **elm** mnohem podrobněji. Nejspíš jí budete chtít pročíst dříve, než začnete **elm** používat.

mailx

mailx(1) je příkazořádkově ovládaný mailový klient. Je velice prostý a nenabízí prakticky nic ve stylu uživatelských rozhraní. Nicméně se hodí ve chvílích, kdy potřebujete rychle něco mailnout, napsat skript na rozesílání většího objemu zpráv, nebo něco takového.

Výchozí podoba příkazové řádky je:

```
$ mailx -s <subject> <to-addr>
```

mailx čte tělo zprávy ze standardního vstupu. Takže můžete text prostě začít psát a až budete hotoví, stiskněte **ctrl+D**. Rovněž můžete předat text mailx-u příkazem cat.

Následuje příklad zaslání souboru se zdrojákem programu nějaké další osobě.

```
$ cat randomfunc.c | mailx -s "Here's that function" asdf@example.net
```

Manuálová stránka vysvětluje více, co **mailx** dokáže, takže asi budete chtít nejdřív kouknout tam, než ho začnete používat.

Předchozí FTP klienti | Výchozí Nahoru | Další lynx

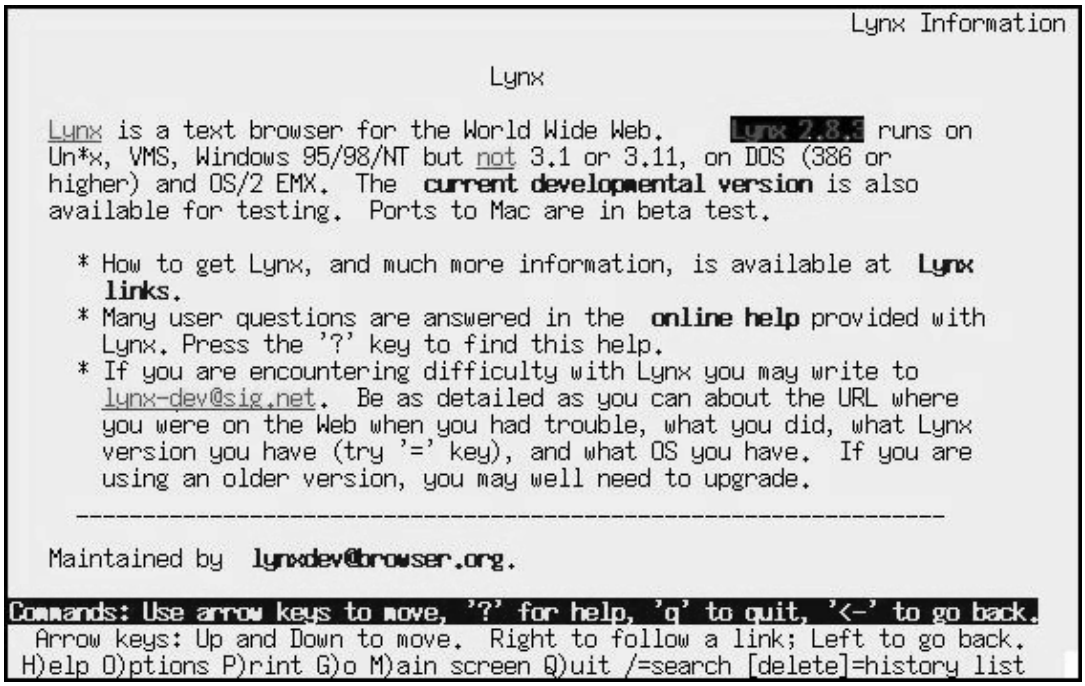
lynx

lynx(1) je textově orientovaný webový browser. Nabízí velmi rychlou cestu, jak se podívat na něco v Internetu. Sometimes graphics just get in the way if you know exactly what you're after.

Pro spuštění programu **lynx** jednoduše napište:

```
$ lynx
```

Obrázek 13–4. Výchozí stránka programu Lynx.



Při spouštění můžete rovnou zadat i jaká stránka se má z Internetu načíst:

```
$ lynx http://www.slackware.com
```

lynx vám zobrazuje příkazové klávesy a co dělají v dolní části obrazovky. Šipky nahoru a dolů posouvají zobrazeným dokumentem, Enter vybírá zvýrazněný odkaz a šipkou vlevo se vrátíte na předchozí navštívenou stránku. Stiskem **d** zahájíte download (stahování) vybraného souboru. Klávesa **g** zobrazí výzvu `Go`, kam můžete zadat URL, které chcete otevřít.

lynx má ještě řádku dalších příkazů. Jednak můžete konzultovat manuálové stránky a nebo stiskem **h** vyvolat nápovědu.

wget

wget(1) je příkazořádková utilita, sloužící ke stahování souborů ze zadaných URL. Je užitečná pro stahování celých webových sajtů pro off−line prohlížení či pro mnohem bezpečnější stahování souborů z HTTP a FTP serverů, misto Netscapu. Základní syntaxe je:

```
$ wget <url>
```

Rovněž můžete zadávat volby. Například tohle stáhne web−sajt Slackwaru:

```
$ wget --recursive http://www.slackware.com
```

wget vytvoří adresář `www.slackware.com` a uloží do něj soubory tak, jak je to na sajtu uděláno.

wget též může stahovat soubory z FTP sajtů. Prostě jen zadejte nějké FTP URL místo HTTP.

wget má mnohem více voleb, které jej činí šikovným pro sajtově zaměřené skripty (mirrorování web sajtů a podobně). Další informace hledejte v manuálové stránce.

traceroute

Slackware nabízí příkaz **traceroute**(8), převzatý z BSD4.4. Je to užitečný nástroj pro diagnostiku sítě. **traceroute** zobrazuje všechny hostitele, kterými paket prochází, když se snaží dosáhnout cíle. Následující příklad vám ukáže, kolik "hops" (přeskoků) jste vzdáleni od web sajtu Slackware:

```
$ traceroute www.slackware.com
```

Zobrazí se každý hostitel, včetně jeho doby odezvy. Následuje příklad výpisu:

```
$ traceroute www.slackware.com
 traceroute to www.slackware.com (204.216.27.13), 30 hops max, 40 byte packets
 1  zuul.tdn (192.168.1.1)  0.409 ms  1.032 ms  0.303 ms
 2  207.171.227.254 (207.171.227.254)  18.218 ms  32.873 ms  32.433 ms
 3  border-sf-2-0-4.sirius.com (205.134.230.254) 15.662 ms 15.731 ms 16.142 ms
 4  pb-nap.crl.net (198.32.128.20)  20.741 ms  23.672 ms  21.378 ms
 5  E0-CRL-SFO-03-E0X0.US.CRL.NET (165.113.55.3) 22.293 ms 21.532 ms 21.29 ms
 6  T1-CDROM-00-EX.US.CRL.NET (165.113.118.2)  24.544 ms  42.955 ms 58.443 ms
 7  www.slackware.com (204.216.27.13)  38.115 ms  53.033 ms  48.328 ms
```

traceroute je podobný příkazu **ping** v tom, že používá ICMP pakety. Příkaz **traceroute** má několik voleb. Standardně je maximální počet přeskoků nastaven na 30, ale můžete jej změnit pomocí volby "–m". Ostatní volby jsou popsány v manuálové stránce.

Povídání si s druhými lidmi

talk

talk(1) umožňuje lidem, aby spolu "chatovali" – klábosili. Rozdělí obrazovku vodorovně na dvě části. K zaslání žádosti o rozhovor někomu jinému použijte tento příkaz:

```
$ talk <osoba> [ttyname]
```

Obrázek 13–5. Dva uživatelé při "talk" seanci.

Zadáte–li pouze uživatelské jméno, předpokládá se, že chat se má odehrát na lokální úrovni, tudíž dotazováni budou pouze lokální uživatelé. Parametr ttyname je potřeba, pokud je uživatel přihlášen vícekrát, aby bylo jasné, na kterém terminálu se mu má výzva zobrazit. Potřebné informace o uživatelích můžete získat pomocí příkazu **w**(1).

talk může kontaktovat i uživatele na vzdálených hostitelích. Jako uživatelské jméno pak použijte e–mailovou adresu. Talk se pokusí kontaktovat uživatele na hostiteli, jak to vyplývá z e–mailové adresy.

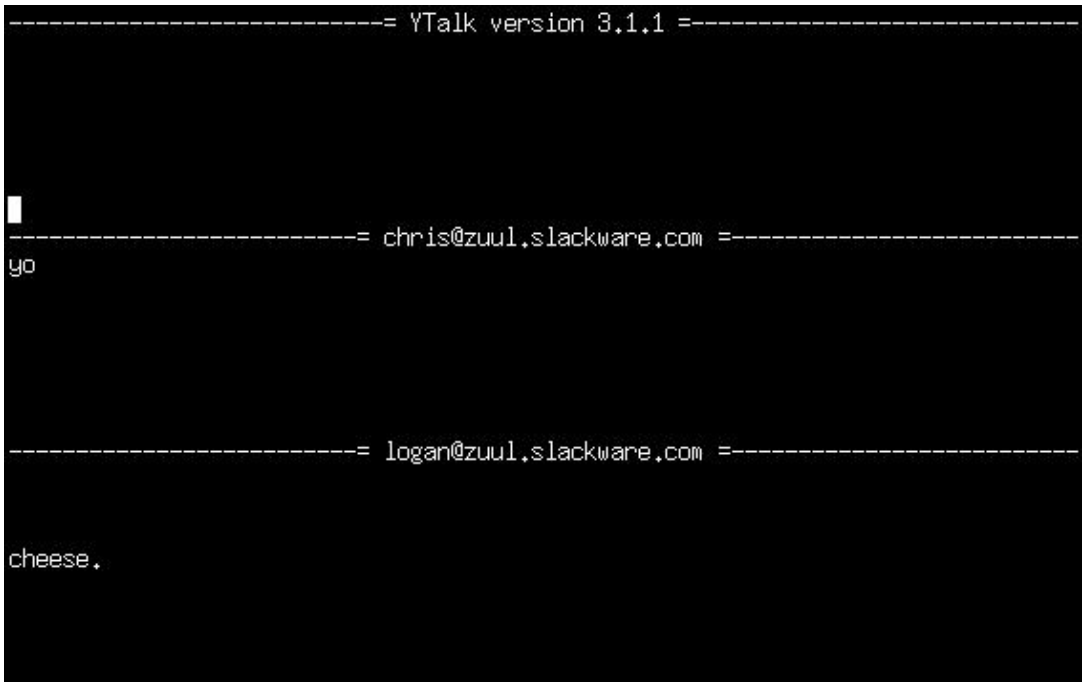
talk má svá omezení. Podporuje rozhovor pouze pro dva uživatele, a to v polo–duplexním režimu.

ytalk

ytalk(1) je náhradou za talk a je s ním dokonce (zpětně) kompatibilní. Ve Slackwaru je spoušten stejnojmenným příkazem **ytalk**. Syntaxe je podobná, ale má pár odlišností:

```
$ ytalk <username>[#ttyname]
```

Obrázek 13–6. Tři uživatelé při ytalk seanci.



Uživatelské jméno a terminál se zadávají stejně jako u **talk**u. Pouze je neoddělujete mezerou, ale znakem hash (#).

ytalk nabízí několik rozšíření:

- Podporuje rozhovor víc než jen dvou uživatelů.
- Nabídku voleb, která může být kdykoliv vyvolána pomocí **Esc.**
- Během rozhovoru si můžete odskočit do shellu.
- A další...

Jste–li administrátorem serveru, měli byste se ujistit, že `/etc/inetd.conf` umožňuje komunikaci přes ntalk port. Bez toho by **ytalk** nepracoval správně.

Shrnutí

Nyní byste měli znát některé základní síťové diagnostické příkazy. Jejich použitím byste měli být schopni rozpoznat, zda problém který nastal je ve vašem počítači, nebo v síti mezi ním a vzdáleným systémem. Také bystě měli vědět o existenci několika e−mailových klientů, webových prohlížečů, ftp klientů a komunikačních programů.

bzip2

bzip2(1) je alternativní kompresní program instalovaný ve Slackware Linuxu. Používá jiný komprimační algoritmus než **gzip**. To vyústí uje do několika výhod a několika nevýhod. Hlavní výhodou **gbzip2** je velikost zkomprimovaného souboru. **bzip2** téměř vždy zkomprimuje soubor lépe, než **gzip**. V některých případech to může dávat podstatně menší soubory. To může být velká výhoda pro lidi, kteří mají pomalejší modemové připojení.

Nevýhodou programu **bzip2** je, že zatěžuje CPU mnohem více než **gzip**. To obecně znamená, že bzipování souboru bude trvat déle a bude více využívat CPU, než kolik by vyžadovalo gzipování. Když se rozhodujete, který kompresní program použít, musíte zvážit rychlost vs. velikost a rozhodnout, co je důležitější.

Používání programu **bzip2** je velmi podobné používání **gzip**u, takže se nebudeme zdržovat jeho popisováním. Jednoduše zadejte **bzip2** a název souboru, který chcete zkomprimovat:

```
$ bzip2 infile
```

Výsledný výstupní soubor bude obvykle menší, než vstupní a bude se jmenovat `infile.bz2`. Tak jako u **gzip**u nebude už vstupní soubor existovat – **bzip2** nahradí vstupní soubor jeho zkomprimovanou kopií.

Stejně jaku u programu **gzip** můžete používat číselné volby pro ladění kompresního poměru a rychlosti. Následující příklad ukazuje, jak dosáhnout maximální komprese (s odpovídajícím vytížením CPU):

```
$ bzip2 -9 infile
```

Pro dekompresi souborů jejichž název končí .bz2 existují dva programy stejně, jako tomu bylo u programu **gzip**. Můžete použít **bzip2** nebo **bunzip2**/(1). Verze s **bzip2** vyžaduje zadání argumentu –d v příkazovém řádku:

```
$ bzip2 -d infile.bz2
```

To dekomprimuje bzipovaný soubor a nahradí jej dekomprimovanou verzí. Název výsledného souboru bude o příponu .bz2 kratší. Podobně můžete k dekompresi bzipovaných souborů používat **bunzip2**:

```
$ bunzip2 infile.bz2
```

Oběma způsoby dosáhneme téhož – opět díky symbolickému linku. Otestování `/bin/bunzip2` ukáže, že je to jednoduše symbolický link na `/bin/bzip2`. Využívá se tu téhož triku jako u **gzip**u. Zjistíte, že volání téhož programu pod různými jmény, aby se dosáhlo odlišného chování, je oblíbená finta linuxových programátorů.

```
$ cd /bin
 $ ls -l bunzip2
lrwxrwxrwx   1 root  root      5 Feb  2 09:45 /bunzip2 -> bzip2
```

tar

tar(1) je páskový archivátor z projektu GNU. Z několika souborů a adresářů vytvoří jeden velký soubor. To pak umožňuje komprimovat celý adresářový strom, což by použitím samotného **gzip** nebo **bzip2** nebylo možné. Program **tar** má množství voleb zadávaných z příkazového řádku. Ty jsou popsány v jeho manuálové stránce. Tato sekce pokrývá pouze nejobvyklejší způsoby jeho využití.

Tím úplně nejčastějším způsobem použití programu **tar** je dekomprese a rozbalení (unarchive) balíčku, který jste si stáhli z webu nebo ftp. Většina těchto souborů má příponu `.tar.gz`. Obvykle se tomu říkává "tarball". Znamená to, že několik souborů bylo zabaleno pomocí **tar**, a potom zkomprimováno **gzip**–em. Rovněž to někdy můžete vidět ve formě `.tar.Z`. To znamená to samé, jen to bylo nejspíš vytvořeno na starším unixovém systému.

Alternativně můžete někde objevit soubor s příponou `.tar.bz2`. Třeba zdrojové balíčky kernelu jsou obvykle distribuovány v této formě, protože je to menší pro download. Jak jste asi uhodli, je to několik souborů sbalených pomocí **tar** a posléze bzipovaných.

Ke všem souborům v archivu se můžete dostat použitím **tar**u s jistými argumenty zadanými na příkazovém řádku. Při rozbalování tarballu používáme příznak **-z**, který říká, aby se před rozbalením použil **gunzip** a dekomprimoval jej. Obená forma dekomprese tarballu je následující:

```
$ tar -xvzf hejaz.tar.gz
```

To je celkem dost voleb. Takže co všechny znamenají? Volba **-x** znamená extract – rozbalit. To je důležité, protože to říká **tar**u, co má se vstupním souborem udělat. V tomto případě bude vstupní soubor rozbalen zpět do všech těch souborů a adresářů, ze kterých byl vytvořen. Volba **-v** znamená, aby byl tar upovídaný (vebose). Takže tar bude vypisovat informace o všech souborech, které z archivu vybaluje. Je zcela přijatelné nechat tuto volbu vypnutou, když se ukáže nudnou. V opačném případě můžete zkusit **-vv** a dosáhnout tak mimořádné upovídanosti a získat výpis s mnoha dalšími informacemi o každém vybalovaném souboru. Voba **-z** říká **tar**u, aby `hejaz.tar.gz` nejprve prohnal **gunzip**em. A konečně volba **-f** říká **tar**u, že následující řetězec je jménem archivního souboru, o jehož rozbalení tu celou dobu běží.

Je několik způsobů, jak zapsat tentýž příkaz. Na starších systémech postrádajících pořádnou kopii GNU taru, můžete zahlédnout tento zápis:

```
$ gzip -dc hejaz.tar.gz | tar -xvf -
```

Tento příkazový řádek unzipuje soubor a výstup pošle do programu **tar**. Volba –c říká, aby **gzip** poslal výsledek své práce do standardního výstupu (stdout), místo do souboru. Rourou se pak tento výstup předá programu **tar** k rozbalení. "–" na konci znamená, že místo souboru se mají data brát ze standardního vstupu (zde z roury). To rozbalí proud dat, který přichází z **gzip**u a zapíše je na disk.

Další způsob jak zapsat první příkaz, je vynechat pomlčku před volbami:

```
$ tar xvzf hejaz.tar.gz
```

Také můžete narazit na bzipované archivy. Verze programu **tar**, dodávaná se Slackware Linuxem je umí obsluhovat stejně jako gzipované archivy. Místo volby **-z** zadejte **-j**:

```
$ tar -xvjf foo.tar.bz2
```

Je důležité si zapamatovat, že **tar** ukládá vybalené soubory do aktuálního adresáře. Takže máte–li nějaký archiv v adresáři `/tmp`, a chcete jej rozbalit do svého domovského adresáře, máte dvě možnosti. Zaprvé můžete archiv přesunout do svého domovského adresáře a rozbalit ho tam. Nebo můžete zadat v příkazovém řádku cestu k archivu:

```
$ tar -xvzf /tmp/bar.tar.gz
```

Obsah archivu se vybalí do vašeho domovského adresáře, zatímco originální archiv zůstane stále v adresáři /tmp.

Vytvoření archivu

Druhou nejobvyklejší činností s programem **tar** je vytváření vaších vlastních archivů. Vytvoření archivu není o mnoho komplikovanější, než jeho rozbalování. Jen se použije jiná sada voleb v příkazovém řádku.

K vytvoření komprimovaeného archivu ze všech souborů v aktuálním adresáři včetně všech podadresářů a souborů v nich obsažených použijete **tar** v této formě:

```
$ tar -cvzf archive.tar.gz .
```

Volba **-c** na příkazovém řádku říká programu **tar**, aby vytvořil archiv. Volba **-z** prožene vytvořený archiv **gzip**em, aby se zkomprimoval. Ta tečka na konci označuje aktuální adresář (co se má archivovat). `archive.tar.gz` je jméno souboru v němž bude archiv uložen. Můžete si ho pojmenovat jak chcete a zdáte–li v názvu i cestu, bude archiv umístěn tam, kam cesta ukazuje. Uveďme si příklad:

```
$ tar -cvzf /tmp/archive.tar.gz .
```

Archiv pak bude vytvořen v `/tmp`. Co se má archivovat můžete zadat jako seznam souborů a adresářů na konci příkazové řádky:

```
$ tar -cvzf /tmp/archive.tar.gz file1 file2 directory1 file3 ...
```

zip

Nakonec tu máme ještě dvě utility používané ke komprimaci souborů. Jsou velmi rozšířené ve světě Windows, takže i Linux je – kvůli přenositelnosti souborů – obsahuje. Tím kompresním programem je **zip**(1) a dekompresním protějškem program **unzip**(1).

Komprese jednoho souboru je snadná:

```
$ zip foo *
```

Takto se vytvoří soubor `foo.zip`, který bude obsahovat všechny soubory, které jsou v aktuálním adresáři. Příponu .zip přidává **zip** automaticky, takže není nutné ji za jméno souboru psát. Zip může do archivu přidávat rekurzivně i adresáře a soubory, které jsou vnořené v aktuálním adresáři pomocí volby –r:

```
$ zip -r foo *
```

Dekomprese souborů je opět snadná:

```
$ unzip foo
```

Takto extrahujete všechny soubory ze souboru `foo.zip`, včetně veškerých adresářů, které by archiv obsahoval. Příponu .zip opět nemusíte zadávat.

Příkazy **zip** mají několik pokročilých voleb, které umožňují vytvářet samorozbalovací archivy, vynechávat soubory, řídit velikost komprimovaných souborů, vypisovat co se přihodí a mnohé další. Na použití těchto voleb se podívejte do manuálových stránek.

Shrnutí

Tato kapitola pojednávala o programech používaných na kompresi, dekompresi a archivování souborů. Měli byste vědět, co to archivní soubor je, jak si ho pomocí **tar** vytvořit a jaké kompresní volby použít. Dále jak archiv dekomprimovat a také jak zacházet s Windowsovatými archivy. Téměř vše, co stahujete nebo nahráváte bude mít formu archivu. Tudíž se jedná o důležité dovednosti.

Módy

Editor **vi** pracuje v různých módech, které se používají k provádění různých úkolů. Když nastartujete **vi**, dostanete se do příkazového módu (command mode). V tomto režimu můžete zadávat různé příkazy pro manipulaci s textem, pohybovat se v textu, ukládat, ukončovat a měnit módy (and mode). Editování textu se provádí v módu vkládání (insert mode). Mezi jednotlivými módy se můžete přepínat snadno pomocí různých kláves, které popíšeme dále.

Příkazový mód (Command Mode)

Na počátku se nacházíte v příkazovém módu. V tomto módu nemůžete přímo vkládat ani upravovat text. Nicméně můžete s textem různě manipulovat, vyhledávat, ukončovat, ukládat, načítat nové soubory atd. To co jsme si teď řekli berte jen jako přičichnutí k příkazovému módu. Jeho různé příkazy budou popsány v sekci *vi klávesy*.

Asi nejpoužívanějším příkazem v příkazovém módu je přepnutí do módu vkládání. Toho se dosáhne stiskem klávesy **i**. Kurzor změní tvar a dole na obrazovce je zobrazen nápis `-- INSERT --` (to se nemusí dít v každém klonu editoru **vi**). Od této chvíle jsou všechny stisky kláves vkládány do aktuálního bufferu a jsou zobrazovány na obrazovce. K návratu do příkazového módu stiskněte klávesu escape.

V příkazovém módu se můžete také pohybovat po souboru. Na některých systémech k tomu můžete použít kláves se šipkami. Na jiných asi budete muset používat tradičnější klávesy "hjkl". Zde je jednoduchý seznam těchto kláves a jejich význam:

h	přesun o jeden znak vlevo
j	přesun o jeden znak dolů
k	přesun o jeden znak nahoru
l	přesun o jeden znak vpravo

Jednoduše stiskněte některou z těchto kláves. Jak uvidíte později, tyto klávesy je možné kombinovat s čísly a pohyb je pak mnohem rychlejší.

Mnoho příkazů, které používáme v příkazovém módu, začíná dvojtečkou. Například pro ukončení programu vi zadáte příkaz **`:q`**. Dvojtečka indikuje, že se jedná o příkaz, zatímco "q" říká editoru, aby se ukončil (quit). Dalšími příkazy jsou (volitelně) číslo následované písmenem. Těmto příkazům dvojtečka nepředchází a jsou obecně používány pro manipulaci s textem.

Například vymazání řádku se provádí stiskem **`dd`**. Tak vymažete řádek, ve kterém je kurzor. Zadáním příkazu **`4dd`** řeknete editoru, aby smazal čtyři řádky (aktuální a ještě tři pod ním). Obecně tím číslem říkáme, kolikrát se má zadaný příkaz zopakovat.

Čísla můžete kombinovat i s klávesami pro pohyb kurzoru a přesouvat se tak o větší počet pozic najednou. Například **`10k`** vás přesune o deset řádků výše.

Příkazový mód lze rovněž využít pro vyjímání a vkládání textu a načtení jiných souborů do aktuálního bufferu.

Výběr textu pro kopírování provádíme klávesou **y** (y=yank). Výběr celé aktuální řádky dosáhnete stiskem **`yy`**, čemuž může předcházet ještě číslo ke zkopírování více řádek najednou. Potom se přesunete do místa, kam se má vybraný text vložit a stisknete **p** (p=paste). Text se vloží za aktuální řádek.

A ještě si řekněme něco o vyhledávání. Příkazový mód nabízí jak prosté vyhledání textu, tak velmi složité příkazy pro vyhledávání a nahrazování, využívající skvělých možností regulárních výrazů. Podrobný popis regulárních výrazů je mimo meze této kapitoly, a proto se tu budeme zabývat jen jednoduchými formami vyhledávání.

Jednoduché hledání vyvoláme stiskem klávesy **/**, následované textem, který chceme najít. Editor pak hledá od pozice kurzoru dál směrem ke konci textu a zastaví se v místě, kde hledaný text najde. Přičemž nemusí jít o shodu celých slov, ale jen části. Například vyhledávání slova "lom" najde nejen samostatné slovo "lom", ale i slova "zlom", "plomba", "lomítko", ap. Je to proto, že ve všech ta slova v sobě kombinaci písmen "lom" obsahují.

Poté, co **vi** nalezl první výskyt hledané fráze, můžete vyvolat další její hledání už pouhým stiskem **/** a enteru. Frázi znovu zapisovat nemusíte.

Můžete vyhledávat i opačným směrem. Uděláte to tak, že místo lomítka použijete klávesu **?**. Například, chcete–li, aby se "lom" hledalo směrem k začátku textu, napíšete **`?lom`**.

Mód vkládání (Insert Mode)

Vkládání a přepisování textu provádníme v módu vkládání. Jak už jsme se zmínili, do módu vkládání se z příkazového módu přepnete stiskem klávesy **i**. Potom vše, co mačkáte na klávesnici se vkládá do aktuálního bufferu. Mód vkládádní ukončíte stiskem klávesy **Esc**. Tím se vrátíte do příkazového módu.

Přepisování textu dosáhnete více způsoby. V příkazovém módu vám stisk klávesy **r** umožní nahradit jeden znak, který se nachází pod kurzorem. Pak jen stiskněte nový znak a ten se objeví místo původního a je to. Pak vás to hned přepne zpět do příkazového módu.

Stiskem **R** se přepnete do přepisovacího módu. Ten vám umožní přepisovat tolik textu, kolik jen chcete. Do příkazového módu se vrátíte jako obvykle stiskem klávesy Escape.

Některé verze editoru vi vám nabízejí mož nost přepínat se mezi módy vkládání a přepisování přímo. Stiskněte klávesu "Insert". To vás přenese do módu vkládání. Jakmile jste v módu vkládání fungují dalš í stisky klávesy "Insert" jako přepínač mezi módy vkládání a přepisování.

Předchozí
vi

Výchozí
Nahoru

Dalš í
Otevírání souborů

Otevírání souborů

Soubor můžete otevřít dvěma způsoby. Jednak při spouštění editoru z příkazové řádky můžete zadat i jméno souboru, který pak bude po nastartování otevřen. Nebo, když už máte vi spuštěný, použitím příkazu ":e" v příkazovém módu. Například `/etc/lilo.conf` otevřete takto:

```
:e /etc/lilo.conf
```

Soubor se načte do bufferu a můžete s ním pracovat. Nebyl-li buffer před otevřením souboru prázdný (měli jste tu otevřený jiný soubor), bude stávající soubor v bufferu nahrazen nově otevřeným. Nebyl-li starý soubor před otevíráním nového uložen, bude **vi** protestovat a otevření neumožní (poslední provedené úpravy ve stávajícím souboru by se ztratily). Pokud vám to nevadí, řeknete to editoru tak, že pro otevření použijete upravený příkaz. **:e!**. Obecně vždycky vykřičníkem můžete přebít případné varovné hlášky editoru.

Někdy při editaci textu něco totálně zmastíte. Pak se vám může hodit možnost znovuotevření (ještě nezmaštěné verze) souboru z disku. Uděláte to příkazem **e!**.

Některé klony editoru **vi** (například **vim**) umožňují mít otevřeno více bufferů současně. Pak například pro otevření souboru `09-vi.sgml` v novém bufferu použijete příkaz:

```
:split 09-vi.sgml
```

Obrazovka se pak vodorovně rozpůlí a nový soubor bude zobrazen v její horní polovině, zatímco ten stávající bude v dolní polovině. Různými příkazy pak můžete toto rozdělení obrazovky měnit. Znalcům editoru Emacs ty příkazy budou povědomé. Nejlepší způsob jak se o těchto příkazech něco dovědět je prostudovat manuálovou stránku vašeho vi klonu. Poznamenejme, že mnoho klonů myšlenku rozdělené obrazovky nepodporují, takže jí nebudete moci využít.

Ukládání souborů

vi nabízí několik způsobů, jak uložit soubor. Chcete–li uložit obsah aktuálního bufferu do souboru `randomness`, napište:

```
:w randomness
```

Jakmile máte soubor jednou uložený, jeho opětovné uložení provedete už pouhým zapsáním **`:w`**. Tím uložíte veškeré provedené změny. Po uložení souboru se vi vrátí do příkazového módu. Chcete–li nejen uložit soubor, ale rovněž ukončit práci s editorem (což je dost častá kombinace), můžete zapsat **`:wq`**. To říká editoru **vi**, aby uložil soubor a ukončil svou činnost.

Někdy možná budete potřebovat uložit soubor, který je označen jako "read–only" (jen pro čtení). Dosáhnete toho přidáním vykřičníku za příkaz k uložení:

```
:w!
:wq!
```

A stejně ještě může nastat situace, kdy nemůžete soubor zapsat ani takto (například, když se pokoušíte upravit soubor, který vlastní jiný uživatel). Pokud se tohle stane, **vi** vám řekne, že soubor nelze uložit. Jediná možnost, jak toho přece jenom dosáhnout je (dost špinavej trik), editovat ho jako root.

Ukončování vi

Jeden ze způsobů jak ukončit **vi** je příkaz `:wq`, který uloží aktuální buffer a skončí. Nebo můžete ukončit editor bez ukládání pomocí `:q` či `:q!`.

Někdy se může stát, že počítač, nebo **vi** zhavaruje. Oba editory (elvis i vim) podnikají kroky k minimalizaci poškození otevřených bufferů. Oba editory v nastavených intervalech ukládají buffer do dočasného souboru, který pojmenovávají stejně jako je jméno otevřeného souboru, pouze na začátek jména přidají tečku. Tím navíc označí soubor jako skrytý.

Editor tento dočasný soubor automaticky odstraní pokaždé, když jeho práce skončí normálně. To znamená, že dočasný soubor tam zůstane právě v případech havárií. Když se pak chystáte znovu editovat tento soubor, editor se vás zeptá, kterou verzi vzít. Ve většině případů bude značná část vaší práce zachráněna. **elvis** vám navíc zašle e–mail, kde vás upozorní na existenci oné záložní kopie.

Konfigurace vi

Každý klon **vi** se dá konfigurovat v mnoha směrech.

Když jste v příkazovém módu, můžete zadávat příkazy, které upravují chování editoru **vi**. V závislosti na vašem editoru můžete aktivovat věci, které činí programování snadnějším (jako je zvýrazňování syntaxe, automatické odsazování aj), vytvářet makra pro úlohy automake, aktivovat textové substituce a další.

Téměř všechny tyto příkazy můžete dát do konfiguračního souboru ve vašem domovském adresáři. **elvis** svá nastavení očekává v souboru `.exrc`, **vim** zase v souboru `.vimrc`. Většina nastavovacích příkazů, které mohou být zadávány v příkazovém módu, může být umístěna v konfiguračním souboru. To zahrnuje nastavovací informace, textové substituce, makra a další.

Probírání všech těchto možností a rozdílů mezi editory je dost komplikovaná záležitost, takže na podrobnosti se podívejte do manuálových (případně webových) stránek vašeho oblíbeného editoru. Některé editory (jako **vim**) mají zabudovanou obsáhlou nápovědu, kterou je možno vyvolat příkazem **`:help`**, nebo něčím podobným. Rovněž si můžete pořídit knihu o tomto editoru, například *Learning the vi Editor* od autorů Lamba a Robbinse z nakladatelství O'Reilly.

Mnoho rozšířených linuxových programů vyvolává standardně na práci s textem editor **vi**. Například editování vašich "crontabs" (viz sekce o cronu – která tu není, pozn.překl.:) spustí standardně **vi**. Pokud se vám **vi** nelíbí a rádi byste aby se místo něj spouštěl jiný editor, budete muset nastavit proměnnou prostředí `VISUAL` na jméno editoru, který preferujete (jak se to dělá jsme popisovali v sekci *Proměnné prostředí* v 8. kapitole. Chcete–li mít jistotu, že váš editor bude nastaven jako výchozí po každém vašem přihlášení do systému, přidejte nastvení `VISUAL` do vašeho konfiguračního souboru pro bash `.bash_profile`, nebo `.bashrc`.

Klávesy vi

Tato část je stručnou referenční příručkou nejpoužívanějších příkazů vi. O některých z nich jsme psali už dříve v této kapitole, zatímco jiné budou nové.

Tabulka 15–1. Přesouvání

Operace	Klávesa
vlevo, dolů, nahoru, doprava	`h, j, k, l`
na konec řádku	`$`
na začátek řádku	`^`
Na konec souboru	`G`
Na začátek souboru	`:1`
Na řádek 47	`:47`

Tabulka 15–2. Editování

Operace	Klávesa
Vyjmutí řádku	`dd`
Vyjmutí pěti řádků	`5dd`
Přepsání znaku	`r`
Vyjmutí znaku	`x`
Vyjmutí deseti znaků	`10x`
Odvolání poslední akce (undo)	`u`
Spojení aktuálního a následujících řádků	`J`

Tabulka 15–3. Vyhledávání

Operace	Klávesy
Vyhledání asdf	`/asdf`
Zpětné hledání asdf	`?asdf`
Opakování posledního hledání vpřed	`/`
Opakování posledního hledání dozadu	`?`

Tabulka 15–4. Ukládání a ukončování

Operace	Klávesa
Ukočení – Quit	`:q`
Ukončení bez ukládání	`:q!`
Ulož it a ukončit	`:wq`
Ulož it, ale nekončit	`:w`
Znovunačíst otevřený soubor	`:e!`
Zapsat buffer do souboru `asdf`	`:w asdf`
Otevřít soubor `hejaz`	`:e hejaz`
Načtení souboru `asdf` do bufferu	`:r asdf`
Načtení příkazu **ls** do bufferu	`:r !ls`

Shrnutí

Nyní byste měli být v základu obeznámeni se standardním unixovým editorem **vi**. Je to vskutku komplexní program s množstvím příkazů a konfiguračních voleb. Vy byste měli znát aspoň jak se soubor otevře, jak se po něm pohybovat, jak ho upravovat, uložit a ukončit práci s **vi**. To jsou ty nejběžnější věci, potřebné při každodenní práci. Jakmile se u vás objeví potřeba silnějších nástrojů, můžete použít nápovědu **vi** a naučit se je odsud.

Balíčovací utility

Pro správu balíčků máme k dispozici čtyři hlavní utility. Ty vykonávají instalaci, odstranění a upgrade balíčků.

pkgtool

pkgtool(8) je nabídkově ovládaný program, který umožňuje instalaci a odstraňování balíčků. Hlavní menu vypadá takto:

Obrázek 16–1. Hlavní menu programu pkgtool.

Je nám nabízeno instalovat z aktuálního adresáře, z jiného adresáře, nebo z disket. Jednoduše vyberte co chcete a **pkgtool** prohledá vybrané umístění a vyhledá balíčky schopné instalace.

Rovněž můžete vidět seznam již nainstalovaých balíčků. Ten vypadá takto:

Obrázek 16–2. Pkgtool – prohlížecí mód.

Pokud chcete odstranit balíčky, vyberte volbu "remove". Bude vám nabídnut zaškrtávací seznam všech nainstalovaých balíčků. Označte ty, které chcete odstranit a dejte OK. **pkgtool** je odstraní.

Někteří uživatelé dávají této utilitě přednost před utilitami pro příkazový řádek. Měli bychom však poznamenat, že utility pro příkazový řádek nabízejí mnohem více možností. Také možnost upgradovat balíčky je nabízena pouze prostřednictvím utilit pro příkazový řádek.

installpkg

Program **installpkg**(8) se stará o instalaci nových balíčků do systému. Jeho syntaxe je:

```
# [ROOT=<path>] installpkg [option] <package name>...
```

installpkg má tři volby, ale jen jednu z nich můžete zadat (ne víc současně).

Tabulka 16–1. volby programu installpkg

Volba	Účinek
–m	Vykoná nad aktuálním adresářem operaci **makepkg**

–warn	Ukáže, co by se stalo, kdybyste nainstalovali zadaný balíček. To je užitečné v produkčních systémech, abyste viděli přesně, co se stane, ještě před tím než spustíte vlastní instalaci.
–r	Rekurzivně nainstaluje všechny balíčky z tohoto a vnořených adresářů. Ve jménech balíčků (<package name>) můžete použít žolíky (wildcards), které budou použity jako vyhledávací maska při rekurzivní instalaci.

Pokud vložíte před **installpkg** proměnnou prostředí ROOT, pak tato cesta bude použita jako root adresář. To je užitečné pro nastavení nových jednotek (drives) pro váš root adresář. Ty jsou typicky mountovány pod `/mnt`, nebo něco jiného než je `/`.

Záznam o nainstalovaném balíčku je vložen do databáze `/var/log/packages`. Tento "záznam" je ve skutečnosti obyčejný textový soubor; jeden pro každý balíček. Obsahuje–li balíček post–instalační skript, ten je zapsán do `/var/log/scripts/<packagename>`.

Můžete zadat více balíčků a taky používat žolíky (wildcards) ve jménech balíčků. Buďte varováni, že **installpkg** vám nebude říkat, že přepisujete nějaký už nainstalovaný balíček. Jednoduše ho nainstaluje na místo toho starého. Chcete–li si být jisti, že staré soubory z předchozího balíčku byly spolehlivě odstraněny, použijte **upgradepkg**.

removepkg

removepkg(8) se stará o odstraňování nainstalovaných balíčků ze systému. Má následující syntaxi:

```
# [ROOT=<path>] removepkg [option] <package name>...
```

removepkg má čtyři možné volby, ale použít můžete vždy jen jednu z nich.

Tabulka 16–2. volby programu removepkg

Volba	Účinky
–copy	Balíček je zkopírován do "preserved packages directory". To vytvoří strom původního balíčku aniž by byl odstraněn.
–keep	Uchová dočasné soubory vytvořené během odstraňování. To je v praxi použitelné jedině pro účely ladění.
–preserve	Balíček je odstraněn, ale současně i zkopírován do "preserved packages directory".
–warn	Ukáže co se stane, když balíček odstraníte.

Pokud vložíte před **removepkg** proměnnou prostředí `ROOT`, pak tato cesta bude použita pro root adresář. To je užitečné pro nastavení nových jednotek (drives) pro váš root adresář. Ty jsou typicky mountovány pod `/mnt`, nebo něco jiného než je `/`.

removepkg prohlíží i ostatní nainstalované balíčky a odstraní jenom ty soubory, které jsou jedninečné pro specifikovaný balíček. Také prohlédne post–instalační skript a odstraní všechny symbolické linky, které jím byly vytvořeny.

Během procesu odstraňování je zobrazována stavová zpráva. Po odstranění balíčku je přesunut záznam z databáze nainstalovaných balíčků do `/var/log/removed_packages` a post–instalační skript je přesunut do `/var/log/removed_scripts`.

Jako u **installpkg** i zde můžete specifikovat několik balíčků, nebo použít žolíky (wildcards) ve jménech balíčků.

upgradepkg

upgradepkg(8) upgraduje nainstalovaný balíček. Má tuto syntaxi:

```
# [ROOT=<path>] upgradepkg <package name>...
```

nebo

```
# [ROOT=<path>] upgradepkg <old package name>%<new package name>
```

upgradepkg nejdříve nainstaluje nový balíček a potom odstraní ten starý, aby se staré soubory nepovalovaly po systému. Pokud se změnilo jméno upgradovaného balíčku, použijte syntaxi se znakem procento k určení starého balíčku (toho nainstalovaného) a nového balíčku.

Pokud vložíte před **upgradepkg** proměnnou prostředí `ROOT`, pak tato cesta bude použita pro root adresář. To je užitečné pro nastavení nových jednotek (drives) pro váš root adresář. Ty jsou typicky mountovány pod `/mnt`, nebo něco jiného než je `/`.

upgradepkg není bez kazů. Vždy byste si měli udělat zálohu konfiguračních souborů. Když se pak přepíší, budete mít kopii originálů pro jakékoliv další opravy.

Pouze u **installpkg** a **removepkg** můžete používat žolíky pro zadání jmen balíčků.

rpm2tgz/rpm2targz

Red Hat Package Manager (RPM) je oblíbený balíčkový systém. Mnoho distributorů softwaru nabízí své produkty v RPM formátu. Jelikož to není náš nativní formát, nedoporučujeme lidem, aby se na něj spoléhali. Bohužel některé věci jsou dostupné jen jako RPM (včetně zdrojů).

Nabízíme program, který zkonvertuje RPM balíčky do našeho nativního formátu .tgz. To vám umožní extrahovat balíčky (třeba pomocí **explodepkg**) do dočasného adresáře a vyzkoušet jejich obsah.

Program **rpm2tgz** vytvoří balíček Slackware s koncovkou .tgz, zatímco **rpm2targz** vytváří archiv s koncovkou .tar.gz.

Vytváření balíčků

Vytváření Slackware balíčků může být i snadné i těžké. Neexistuje žádná specifická metoda pro sestavení balíčku. Jediným požadavkem je, aby byl balíček gzipovaný tar soubor a obsahuje–li post–instalační skript, aby tento byl v souboru `/install/doinst.sh`.

Pokud se zajímáte o vytváření balíčků pro váš systém nebo pro síť kterou spravujete, měli byste se podívat do různých sestavovacích skriptů ve zdrojovém stromu Slackwaru. Je několik metod, které používáme pro vytváření balíčků.

explodepkg

explodepkg(8) udělá tu samou věc jako **installpkg –** rozbalí balíček, ale ve skutečnosti jej nenainstaluje a neudělá záznam v databázi balíčků. Jenom balíček rozbali do aktuálního adresáře.

Podíváte–li se do zdrojového stromu Slackwaru, uvidíte jak využíváme tohoto příkazu v "strukturálních" balíčcích. Tyto balíčky obsahují kostru toho, jak bude vypadat finální balíček. Obsahují všechny nezbytné názvy souborů (s nulovou velikostí), práva a vlastnictví. Sestavovací skript pomocí **cat** přenese obsah ze zdrojového adresáře do adresáře pro sestavování balíčků.

makepkg

makepkg(8) sbalí aktuální adresář do platného Slackware balíčku. Prohledá strom, jestli tam jsou symbolické linky a přidá vytvářecí blok do post–instalačního skriptu, který tyto linky vytvoří během instalace balíčku. Rovněž upozorňuje na soubory s nulovou délkou, vyskytnou–li se ve stromu balíčku.

Tento příkaz se typicky spouští poté, co jste si vytvořili váš balíčkový strom.

Vytváření Tagů a Tag souborů (pro setup)

Program setup obsluhuje instalaci softwarových balíčků ve vaš em Slackware systému. Při práci využ ívá tag soubory, které mu říkají, které balíčky musí být nainstalovány, které jsou volitelné a které jsou setupem vybrány jako defaultní.

Tag soubor najdete v adresáři kaž dé softwarové skupiny pod názvem `tagfile`. Obsahuje seznam balíčků v této skupině a jejich status. Status můž e mít tyto hodnoty:

Tabulka 16–3. Mož nosti statutu balíčku v tagfile.

Volba	Význam
ADD	add – přidat: Balíček je vyž adován pro správnou funkci systému.
SKP	skip – přeskočit: Balíček bude automaticky vynechán.
REC	recommended – doporučený: Balíček není nutně vyž adován, ale doporučuje se.
OPT	optional – volitelný: Instalaci takového balíčku si rozhodněnte zcela sami.

Formát je jednoduchý:

```
<jméno balíčku>: <status>
```

Jeden balíček na řádek. Originální tag soubory pro kaž dou softwarovou skupinu jsou ulož eny jako `tagfile.org`. Proto, když si zpackáte svůj vlastní `tagfile`, můž ete si jej z tohoto `tagfile.org` obnovit.

Mnoho administrátorů dává přednost vytváření vlastních tag–souborů a spouš tějí installer s volbou "full". Program setup pak čte tyto tag–soubory a vykonává instalaci v souladu s jejich obsahem. Pokud je použ it status REC nebo OPT, zobrazí se dialogové okno s dotazem, zda balíček s tímto statutem instalovat či nikoliv. Je proto doporučováno, abyste si tag–soubory pro automatickou instalaci vyplnili předevš ím (nebo výhradně) statuty ADD a SKP.

Hlavně si dejte pozor, jestli máte své tag–soubory zapsané v tom samém adresáři, kde jsou originály. To je samozřejmě problém, jsou–li originály na CD, na které se nedá zapsat. Na tuto situaci Slackware setup pamatuje a nabízí mož nost číst tag–soubory i z jiného, alternativního umístění (adresáře či diskety).
Pozn.překl: Chcete–li této mož nosti využ ít, vytvořte si v alternativní lokaci kompletní adresářový strom programových skupin a do adresářů nahrajte odpovídající tag–soubory s vaš imi úpravami. Pokud nechcete upravovat vš echno, udělejte si tag–soubory jen pro ty skupiny, které chcete změnit oproti originálu. Když setup nenajde tag–soubor v alternativní lokaci, použ ije originální z adresáře skupiny na zdrojovém instalačním médiu. Chytrý, co? :o)

Shrnutí

Nyní byste měli chápat ideu softwarových balíčků a jak je realizována ve Slackwaru. Měli byste ovládat různé utility pro správu balíčků. Nejdůležitější část této kapitoly je o tom jak nainstalovat, odstranit a upgradovat balíčky. To je nejčastější využití balíčkových utilit. Také byste měli mít určité povědomí o vytváření a prohlížení balíčků.

Obstarání ZipSlacku/BigSlacku

Obstarání ZipSlacku nebo BigSlacku je jednoduché. Jestliže jste zakoupili oficiální Slackware Linux CD set, tak už máte ZipSlack a BigSlack. Najděte CD obsahující ten který chcete, a vložte ho do CD mechaniky. Je to obvykle ten třetí ze čtyř disků, ale spíš věřte nálepce než této dokumentaci.

Jestliže chcete ZipSlack nebo BigSlack stáhnout z Internetu, nejprve byste měli navštívit naší Get Slack stránku, kde jsou nejčerstvější informace:

http://www.slackware.com/getslack/

ZipSlack a BigSlack jsou součástí každého Slackware vydání. Najděte vydání, které chcete, a vstupte do jeho adresáře na FTP serveru. Adresář posledního vydání je k nalezení na této adrese:

ftp://ftp.slackware.com/pub/slackware/slackware/

ZipSlack najdete v podadresáři `/zipslack` a BigSlack v `/bigslack`. ZipSlack je nabízen jako jeden velký .ZIP soubor, nebo po částech, které se vejdou na disketu. Tyto části jsou v adresáři `/zipslack/split`. BigSlack je nabízen jen po částech.

Nestahujte pouze .ZIP soubory. Měli byste si také stáhnout dokumentaci a všechny boot image, které jsou v adresáři.

Instalace

Poté, co jste si obstarali nezbytné soubory, rozbalte .ZIP soubor (nebo soubory stažené po částech). Je potřeba použít 32−bitový archivační software. Velikost a jména souborů v archivu jsou pro 16−bitový těžkým oříškem. Použijte např. WinZip nebo PKZIP pro Windows.

ZipSlack i BigSlack jsou navrženy pro rozbalení přímo do kořenového adresáře jednotky (třeba C: nebo D:). Vytvoří se `\LINUX` adresář, který obsahuje instalaci Slackwaru. Také zde najdete soubory potřebné k bootování systému.

Poté, co jste rozbalili soubory, měli byste mít na disku, který jste vybrali (jako příklad budeme používat dále C:) adresář `\LINUX`

Bootování ZipSlacku/BigSlacku

Existuje několik možností, jak nabootovat ZipSlack a BigSlack. Nejobvyklejší je použít přiložených `LINUX.BAT` souborů k nabootování systému z DOSu (nebo DOS modu pod Windows 9x). Tento soubor je potřeba editovat tak, aby odpovídal vašemu systému. Pak začne fungovat

Začněte otevřením souboru `C:\LINUX\LINUX.BAT` ve vašem oblíbeném textovém editoru. Na začátku souboru objevíte dlouhý komentář. Tam je popsáno, co musíte změnit v tomto souboru (a také co udělat když budete bootovat z externí Zip jednorky). Nepropadejte panice, když nerozumíte nastavení root= ... . Je tam několik příkladů, tak se nezdráhejte jeden z nich vybrat a vyzkoušet. Jestliže nefunguje, můžete editovat soubor znovu a vybrat jiný.

Poté co učiníte nějaký řádek funkční smazáním rem na začátku řádku, soubor uložte a ukončete editor. Přivedte počítač do DOSu.

> DOS prompt okno ve Windows 9x nebude fungovat!

Napište **C:\LINUX\LINUX.BAT** , abyste nabootovali systém. Jestliže vše jde dobře, měli byste vidět login prompt.

Přihlaste se jako **root**, bez hesla. Pravděpodobně budete chtít pro roota heslo nastavit, a také přidat další účet pro vás. V tomto se obratte na ostatní sekce v této příručce, týkající se obecného použití systému.

Jestliže vám neš el použít soubor `LINUX.BAT` k nabootování systému, přečtěte si soubor `C:\LINUX\README.1ST` , kde jsou popsány další možnosti jak nabootovat.

Přidávání, odstraňování a aktualizace softwaru

ZipSlack a BigSlack mohou používat stejné balíčky jako standardní Slackware instalace. To znamená, že můžete použít standardní balíčkové nástroje Slackwaru k přidávání, odstraňování a aktualizaci softwaru. Můžete balíčky přidávat dokonce přímo z Slackware CD–ROMu.

Další informace hledejte v Kapitole 16

Běžné problémy

Toto jsounejobvyklejší problémy, které uživatele ZipSlacku a BigSlacku mohou potkat. Nabízíme několik dalších možností pomoci, jestliže váš problém zde není uveden. Následující sekce je detailně uvádí.

Unable to open initial console

Toto je většinou způsobeno specifikováním špatné device partition v root= nastavení v souboru `LINUX.BAT` . Editujte znovu soubor `LINUX.BAT` a vyberte jinou root= partition. Jestliže nemáte žádné tuší co tohle znamená, můžete prostě vyzkoušet každou, až natrefíte na tu, co bude fungovat.

Může to také být způsobeno rozbalením .ZIP souborů v jiném místě než kořenovém adresáři jednotky. Archiv musí být rozbalem přímo a ne do podadresáře.

Kernel panic: VFS : Unable to mount root fs

Tohle znamená, že jste specifikovali špatný device pro root= nastavení v souboru `LINUX.BAT` Budete muset editovat tento soubor znovu a vybrat jiný root= device. Jestliže nevíte který to je, prosté zkuste všechny, až najdete ten, který funguje.

Zdroje pomoci

Předtím, než požádáte o pomoc, byste si měli přečíst všechnu dokumentaci v adresáři `C:\LINUX`, třeba je odpověď na váš problém tam.

Jestliže jste si dokumentaci přečetli a problém stále trvá, metody níže popsané vám snad pomohou k jeho vyřešení.

ZipSlack FAQ

Seznam obvyklých dotazů a odpovědí na ně je v souboru FAQ.TXT v adresáři `C:\LINUX` a také na Internetové stránce:

http://www.slackware.com/faq/

ZipSlack diskuzní forum

V online diskuzním foru pro ZipSlack můžete promluvit s dalšími ZipSlack a BigSlack uživateli. Můžete sem poslat svůj dotaz a také nabídnout pomoc ostatním uživatelům. Je to skvělá cesta k výměně zkušeností s ostatními.

http://www.slackware.com/forum/

Podpora přes email

Slackware Support Team se vám pokusí pomoci jestliže máte problémy s ZipSlackem nebo BigSlackem. Zkuste svůj problé specifikovat co nejpřesněji a dodat všechny informace, o kterých si myslíte, že mohou být potřebné k vyřešení problému.

<support@slackware.com>

Shrnutí

Nyní byste měli rozumět tomu, co to je ZipSlack a BigSlack. Jestliže jste se rozhodli pro použití jednoho z nich, měli byste vědět, jak ho nainstalovat, bootovat, odstranit problémy, a jak získat pomoc. ZipSlack a BigSlack mohou být velice příjemné, jestliže chcete Slackware pouze zkusit, ale nechce se vám rušit existující Windows oddíly.

Základy Slackware Linuxu

Oficiální průvodce Slackware Linuxem

David Cantrell

Logan Johnson

Chris Lumens

překlad: Viktor Matys

Obsah

Další
Předmluva