РЪКОВОДСТВО ЗА

LINUX

Том 1

ОСНОВИ НА ИНСТАЛИРАНЕТО И СИСТЕМНА АДМИНИСТРАЦИЯ

МАТ УЕЛШ,
МАТИАС ДАЛХАЙМЕР и ЛАР КАУФМАН

Running Linux, Third Edition
by Matt Welsh, Matthias Kalle Dalheimer and Lar Kaufman

Ръководство за Linux,
Том I - Основи на инсталирането и системна администрация
от Мат Уелш, Матиас Далхаймер и Лар Кауфман

Изданието на български език е публикувано от издателство СофтПрес ООД, 2000
ISBN 954-685-079-9

Авторски колектив:

Редактор: *Мирослава Хънтева*
Прводпечатна подготовка: *Мирослава Хънтева, Гергана Митева*
Художествен редактор*:Димитър Митев*

Издателство за компютърна литература:
СофтПрес ООД
София1407,П.К.114
тел.: (02) 958 25 80, 958 25 67; факс: (02) 58 62 04
e-mail: clients@soft-press.com; Web **site:** www.soft-press.com

За дистрибуция:

СофтПрес София - ул. "Искърско шосе" 19;
тел.: 02/ 973 15 06; e**-mail:** iliev@soft-press.com

СофтПрес Пловдив - бул. "Руски" 139, стая 104;
тел.: 032/ 62 75 62; тел./факс: 032/ 62 27 47; e**-mail:** plovdiv@soft-press.com

СофтПрес Бургас - пл. "Тройката" 4;
тел.: 056/ 80 02 31; факс: 056/ 80 31 39; e**-mail:** burgas@soft-press.com

СофтПрес Стара Загора - ул. "Цар Симеон Велики" 117;
тел.: 042/ 60 27 75; e**-mail:** stzagora@soft-press.com

КРАТКО СЪДЪРЖАНИЕ

ТОМ 1

ОСНОВИ НА ИНСТАЛИРАНЕТО И СИСТЕМНА АДМИНИСТРАЦИЯ

ТОМ II
ПРИЛОЖЕН СОФТУЕР И ТЕМИ ЗА НАПРЕДНАЛИ

СЪДЪРЖАНИЕ НА ТОМ I

ПРЕДГОВОР

Linux е безплатна реализация на Unix за Intel 386 (или по-добър процесор), Alpha, SPARC, PowerPC, Motorola 680x0 (m68k) и други персонални компютри и сървърни архитектури. В тази книга искаме да ви покажем как можете изцяло да промените начина, по който работите с компютри, като изследваме света на една мощна и безплатна операционна система. Linux се развива различно от традиционния свят на персоналните компютри. Настройката му и работата с него може да бъде предизвикателна, удовлетворяваща и много приятна; смятаме, че Linux връща голяма част от удоволствието при работата с компютър. Каним ви да се потопите в книгата, да й се насладите и да станете първия човек в квартала, който знае как да използва синхронизиращите честоти на монитора и как да настрои ядрото с *rdev*.

Тази книга е насочена към читатели, които са любопитни и достатъчно съзидателни, за да се заровят с всичка сила в света на Linux. Linux е нещо като въстание срещу платените операционни системи и много от потребителите му са хора, които обичат да живеят на ръба на последните технологични тенденции. Разбира се, случайният читател може да инсталира и използва Linux (или дори стотици Linux-и!) без особени трудности, но целта на книгата е да изследва системата в дълбочина и да ви покаже начина на мислене в нея. Вместо да коментираме объркващи детайли, ще ви обясним принципите, по които работи Linux, така че да можете сами да отстранявате евентуалните проблеми. В тази книга използваме знанията на няколко експерти по Linux и се надяваме да ви научим на много неща, така че един ден да можете спокойно да се наречете истински Linux гуру.

Това е третото издание на книгата. В него изцяло сме обновили главите за инсталиране и конфигуриране, така че да съответстват на последните дистрибуции на Linux (включително Red Hat, SuSE и Debian) и на много от приложните пакети. Ядрото на книгата обаче не е много изменено. Това е умишлено: в първите две издания положихме големи усилия да направим книгата колкото е възможно по-ясна, независимо че Linux непрекъснато се развива. Тази философия работи по забележителен начин и сме я запазили и в това ново, обновено издание. Надяваме се книгата да ви бъде полезна дълго време.

Разглеждаме предимно версията на Linux за процесорната фамилия Intel x86 (чиповете 386/486, Pentium, Pentium Pro и Pentium ИЛИ). Включили сме приложения, в които се обяснява подробно инсталирането и основните процедури по настройката на Linux и за няколко други компютърни архитектури, включително Alpha, SPARC, m68k и PowerPC. С изключение на частта, отнасяща се до основните инсталационни процедури, книгата е приложима за Linux върху произволна платформа.

Матиас Далхаймер се присъедини като съавтор към Мат Уелш и Лар Кауфман за това трето издание. Матиас беше известно време деен член на Linux-обществото и сътрудник в разработването на KDE (разпространена графична работна среда за Linux). Той написа повечето от новия материал в това издание и неговите професионални умения създадоха нови перспективи за техническата част на книгата.

В предговора към първото издание отбелязахме, че Linux има потенциал изцяло да промени образа на света от операционни системи за персонални компютри. Като се вглеждаме назад, виждаме, че това вече се случи. Linux нахлу в компютърните среди с удивителна сила - той се коментира от основните медийни канали, помогна за обявяването на революцията, наречена "Отворен Код" и претендира да е най-жизнения конкурент на Microsoft на пазара за операционни системи. Днес броят на потребителите на Linux по света се оценява на над 10 милиона. Linux се разви до състояние, в което за повечето хора не е задължително да знаят много за системата си, за да работят с нея. Те могат просто да си инсталират програмите и да ги използват. Така че, вероятно ще сметнете много от подробните съвети в тази книга за ненужни. В такъв случай можете да ги използвате като исторически факти.

Пишейки тази книга, искахме да направим Linux реален избор за много от потребителите на персонални компютри, на които пречат ограниченията на платените операционни системи. Понеже Linux е създаван от много хора, някои негови аспекти са донякъде объркващи или очевидно уникални. В тази книга се опитваме да кондензираме колкото е възмож-

но повече полезна информация, като използваме кореспонденцията си с хиляди от потребителите на Linux по света и опита, който сме придобили при работата с нашите собствени Linux системи. Linux може да промени начина, по който мислите за компютрите и тази книга ще ви покаже как.

Светът на Linux се промени много за времето от предишното издание. Но повечето от промените са в неговия образ, а не в същността му: Forbes Magazine и Economist са впечатлени от ръста на Linux, основни компютърни компании като IBM, Hewlett Packard, Oracle и Compaq поддържат Linux по различни начини, а моделът "Отворен Код" (познат в началото като безплатен софтуер), за който Linux служи като пример, е непрекъснато в новините.

Самият Linux се подобрява през цялото време и програмите за него имат впечатляващо развитие. Днес Linux може да се използва за солидни сървъри с много процесори, клъстери и RAID-масиви от дискове, които изпълняват важни задачи. Това е уводна книга и затова тук не се спираме на тези възможности. Вместо това се концентрираме върху основните възможности на системата, като все пак изследваме няколко теми, свързани със сървърите (по-конкретно, настройката на Samba и web-сървъри).

Основна нова разработка, която се обсъжда в това издание, е KDE. KDE (и подобният на него проект GNOME) предлага нова, модерна графична работна среда за Linux, подготвяйки системата за завладяването на пазара за настолни операционни системи веднъж и завинаги.

В добавка към новите глави за KDE и Samba, това издание съдържа въведение в PPP и в създаването на Java програми под Linux. Разбира се, цялата останала информация също изцяло е обновена.

Защо хората харесват Linux

Защо всъщност бихте искали да използвате Linux? Това е добър въпрос.
Нали това, което имате в момента, работи. Windows 98 е добра операционна система, но има много ограничения. Понеже е направена за домашни потребители с ниски изисквания, Windows не предоставя производителността и гъвкавостта, която много хора очакват от една операционна система за персонални компютри. Ето няколко причини, поради които хората се прехвърлят на Linux:

Предговор

Той е безплатен.

Linux е безплатно разпространяван вариант на операционната система Unix. Можете да вземете Linux безплатно от някой, който вече го има, от Интернет или да го закупите на разумна цена на компактдиск от множество фирми (понякога в комплект с някои платени програми), като в последния случай често получавате право на безплатна поддръжка.

Той е популярен.

Linux работи на евтини компютри с процесори Pentium Pro, Pentium II, AMD и Cyrix, а дори и на старите 386/486. Освен това, можете да го използвате на работни станции от висок клас, базирани на архитектурите SPARC и Alpha, на PowerPC и на 68k базирани Макове. Linux поддържа голям спектър видео и звукови платки, компактдискови устройства и твърди дискове, принтери, скенери и други устройства.

Linux има огромно общество от потребители, представено в Интернет с много web-страници, посветени на предоставянето на информация и обсъждането на системата. Все повече производители на платен софтуер разработват приложения за Linux. Заслужава си да споменем WordPerfect на Corel, пакета Star Office на Star Division (вече част от Sun Microsystems) и някои продукти за работа с бази данни на производители като Oracle, Informix и IBM.

Той е мощен.

Ще бъдете учудени колко бързо работи системата, дори при много едновременно работещи процеси и много отворени прозорци. Linux използва хардуера по превъзходен начин. Много от платените операционни системи (включително Windows 98) почти не използват съвременните многозадачни възможности на Pentium. Linux е създаден за тази архитектура и я използва докрай. Компютър с Linux, бърз процесор и достатъчно количество памет може да работи като (или дори по-добре от) Unix-работни станции, струващи по 10 000 или повече щатски долара.

Той е с добро качество и поддържа висококачествени приложения.

Linux е разработен открито, като стотици програмисти и потребители са го усъвършенствали, но под личното наблюдение и с напътствия от създателя му, Линус Торвалдс. Linux включва разработки на университети, фирми и отделни личности във формата на добре разработени компилатори, редактори и други програми и скриптове, създавани през последния четвърт век. За разлика от другите нови операционни системи, за Linux има огромна количество програми,

които можете свободно да използвате - от основни научни приложения до мултимедийни програми и игри.

Той има всички възможности на Unix.

Linux е истинска многопотребителска и многозадачна операционна система, която поддържа много процесори на някои платформи. Той използва графичния потребителски интерфейс на системата X Window и предлага няколко интегрирани графични работни среди, в това число KDE и GNOME. Linux предоставя пълна поддръжка на всички важни мрежови протоколи, включително TCP/IP, SLIP, PPP и UUCP.

Той е изцяло съвместим с Windows 95/98, Windows NT и MS-DOS.

Можете да инсталирате Linux на твърд диск, който съдържа Windows 98 или друга операционна система. Linux може директно да използва файлове от Windows-частта на диска или от дискета. Разработва се емулатор за Windows 98 и MS-DOS, така че вероятно ще можете да използвате в Linux платени приложения, създадени за Windows. Linux не работи под Windows - той е изцяло независим от него, но съдържа софтуер, които позволява на двете системи да работят съвместно.

Той е малък.

Основната система (включително графичната потребителска среда) може да работи дори с 8 MB памет. Система с основните възможности ще се събере в 10 до 20 MB дисково пространство. (Ако това ви изглежда много, то е защото Linux съдържа много полезни програми). Linux може да работи във вградени системи с малко памет (като тези, които се използват в мрежовите маршрутизатори и роботи), и дори в джобни устройства като Palm-Pilot!

Той е голям.

Някои от съвременните дистрибуции запълват повече от 1 GB некомпресирано дисково пространство само за изпълнимите файлове. (Пълният изходен текст е достъпен безплатно и е включен в много дистрибуции на софтуера на компакт-диск.) Количеството на адаптираните за Linux мощни приложения расте непрекъснато. Вероятно бихте могли да запълните няколко гигабайта с тях, дори без графичните и звуковите файлове. Linux се използва за създаването на едни от най-големите супер-компютри в света - в мрежа се свързват много персонални компютри, като на всеки от тях работи Linux и всички те се използват като един голям компютър.

Повечето потребители на Linux могат да използват система, заемаща около 300 MB дисково пространство. Това включва всички основни програми, както и приятни допълнения като графичната среда на системата X Window, приложения за текстообработка и системи за разработка на софтуер, като компилатори и библиотеки. А ако наистина сте умел потребител, можете да използвате още много други приложения.

Той се поддържа.

Най-голямата поддръжка са множеството Интернет-страници, посветени на Linux (заедно с хилядите участници в дискусиите по Интернет). Освен това, можете да се абонирате за поддръжка при независима компания или да закупите поддържана версия на Linux от някой от разпространителите му.

Той е добре документиран.

Тази книга е едно чудесно начало и е преведена на френски, немски, японски и китайски език. Преди време обществото на разработчиците на Linux основа Linux Documentation Project (LDP - проект за документиране на Linux), който осигурява голямо количество електронна документация за системата. Многото книги, списъци с често задавани въпроси и документи от типа "как да ..." от LDP могат да ви помогнат да се справите с почти всяка задача, която трябва да решите под Linux. След като преодолеете началните инсталационни проблеми, Linux е повече или по-малко като всяка друга Unix система, така че много от общите книги за използване и администриране на Unix ще ви помогнат. Освен това, съществуват стотици книги специално за Linux - встъпителни и за напреднали - които са преведени на повечето основни езици по света.

Организация на книгата

Всяка глава в това издание съдържа голямо количество информация. Тя ви въвежда в материал, който лесно би могъл да запълни няколко книги. Ние обаче ще преминаваме бързо през темите, които трябва да знаете.

Глава 1, *Въведение в Linux,* се опитва да сближи много различни теми. Тя обяснява защо възникна Linux и какво го различава от другите версии на Unix и от операционните системи за персонални компютри.

Глава *2, Подготовка за инсталиране на Linux,* описва действията, които трябва да извършите преди инсталиране, например разделянето на твър-

дия диск на части (в случай, че искате да използвате и друга операционна система освен Linux).

Глава 3, *Инсталиране и начално конфигуриране,* е изчерпателно ръководство за инсталиране и настройка на Linux за вашата система.

Глава 4, *Основни команди и концепции в Unix,* предлага въведение в Unix за системния администратор. Целта й е да ви даде достатъчно знания, за да се справите с основните задачи, които са описани в тази книга. Обяснени са основните команди заедно с някои съвети за администратори и концепции, които трябва да знаете.

Глава 5, *Основи на системната администрация;* Глава 6, *Управление на файловите системи, виртуалната памет и файловете на хардуерните устройства;* Глава **7,** *Актуализиране на софтуера и ядрото;* и Глава 8, *Други административни задачи,* описват системната администрация и поддръжка. Това са може би най-важните и най-полезни глави в книгата; те описват управлението на потребителски профили, създаването на резервни копия, как се актуализира софтуерът, създаването на ново ядро и други.

Глава 9, Редактори, текстови инструменти, графика и печат, ви запознават с най-разпространените и често използвани текстови редактори и приложения за Linux - *vi* и Emacs, показват ви как да отпечатате документ и как да използвате различните графични програми.

Глава 10, Инсталиране на системата X Window, ви показва как да инсталирате и настроите системата X Window - мощен графичен интерфейс за Linux и Unix системи. Показваме ви как да преодолеете проблемите, които може би ще срещнете, когато вашата дистрибуция инсталира софтуера и как да го конфигурирате, за да постигнете най-добрата производителност за вашия видео хардуер.

Глава 11, Настройка на графичната среда, ви показва как да създадете ваш собствен графичен интерфейс под системата X Window и обхваща голяма част от всевъзможните настройки, които има X, работната среда KDE и няколко полезни програми, които работят иод X.

Глава 12, Съвместимост с Windows и Samba, представя различните инс-

трумент за взаимодействие с DOS и Windows системи, като се набляга на сървъра SAMBA,който интегрира Linux с дру
Глава 13, Езици за програмиране и Глава 14,Инструменти за програмисти,са за програмисти. Представени са компилат

Глава 15, *TCP/IP и PPP,* ви показва как да настроите тези важни протоколи за връзка с външния свят. Ще научите как да конфигурирате системата си, така че тя да работи в локална мрежа или да се свърже с доставчик на Интернет, като използва PPP.

Глава 16, *World Wide Web и електронна поща,* ви показва как да настроите системата за електронна поща, програмите за четенето ѝ Elm и Netscape Messenger и дори как да пуснете собствен Web сървър.

Приложение А, *Източници на информация за Linux,* ви посочва друга полезна документация за Linux и източници на помощна информация.

Приложение Б, *Проектът GNOME,* съдържа информация за проекта GNOME.

Приложение В, *Инсталиране на Linux върху системи Digital/Compaq Alpha,* ви показва как да инсталирате Linux на първата различна от Intel платформа, за която е реализиран Linux - 64 битовата машина Digital Alpha.

Приложение Г, *Linux PPC: инсталиране на Linux върху компютри PowerPC,* ви показва как да инсталирате Linux на популярната платформа PowerPC.

Приложение Д, *Инсталиране на Linux върху системи с процесор Motorola 680x0,* ви показва как да инсталирате Linux на платформи, които използват процесора на Motorola 680x0 (m68k) като Amiga, Atari и някои модели на Apple Macintosh.

Приложение Е, *Инсталиране на Linux върху системи Sun SPARC,* ви показва как да инсталирате Linux на мощните платформи Sun SPARC.

Приложение Ж, *Параметри на LILO за зареждане на операционна система,* изброява и описва тези параметри.

Приложение З, *Прехвърляне на файлове с протокола Zmodem,* описва инструментите за телекомуникация при използване на модем.

Библиографията съдържа списък с голямо количество книги, документи HOWTO и RFC, които са полезни за повечето потребители и администратори на Linux.

Конвенции използвани в книгата

Следва списък с полиграфските конвенции, които се използват в тази книга:

Удебелен текст	се използва за име на машина, host, потребителски имена и идентификатори.
Наклонен текст	се използва за имена на файлове, директории, програми и команди, опции на команди, e-mail адреси, път към файл, имена на сайтове в Интернет и нови термини.
`текст с постоянна ширина`	се използва в примерите, за да покаже съдържанието на файловете или резултата от изпълнението на команди, за да укаже променливи от обкръжението на програма и ключови думи в кода и за команди на Emacs.
`Текст с постоянна ширина и наклонени букви`	се използва, за да отдели опции, ключови думи или текст, който потребителят трябва да замени с реална стойност.
`Текст с постоянна шнрина и удебепени букви`	се използва в примерите, за да покаже команди или друг текст, който потребителят трябва да напише дословно.

Иконата с бомба в полето отляво е за внимание - тук може да сбъркате така, че да повредите системата или да е трудно да възстановите промените.

Иконата с книга е препратка към друга част на книгата или към друг източник на информация. Цитираният текст под картинката може да е номер или ключ за книгата, документ HOWTO или RFC, споменат в Библиографията, указател към друга глава или приложение на книгата, или препратка към справочната страница на конкретна програма.

Коментари и въпроси

Моля, изпращайте вашите коментари и въпроси отнасящи се за тази книга на оригиналния издател:

O'Reilly & Associates
101 Morris Street
Sebastopol, CA 95472
800-998-9938 (в САЩ и Канада)
707-829-0515 (международен или локален телефон)
707-829-0104 (факс)

Можете да ни изпращате и електронни съобщения. За да се включите в нашия mailing list или за да поръчате каталог, изпратете e-mail до:

n u ts @ o re illy, com

За да зададете технически въпрос или за коментар върху книгата, изпратете e-mail до:

bookquestions @ oreilly. com

Благодарности

Тази книга е резултат от усилията на много хора, и както би могло да се очаква, невъзможно е тук да изброим всички. На първо място, искаме да благодарим на Andy Oram, който извърши превъзходна работа по редактирането, стилизирането и оформлението на книгата, така че тя да добие сегашната си форма. Освен като редактор, Andy ни сътрудничи в главата с ръководството за Unix, както и с материала за X и Perl. Andy беше този, който се обърна към нас, за да напишем тази книга, и той демонстрира търпение на светец, докато чакаше да завършим корекциите си.

Тези от вас, които вече познават Linux, могат да забележат, че някои части от тази книга, като уводната глава и главите за инсталиране на Linux, са публикувани като част от *Linux Installation and Getting Started* - безплатна книга, която можете да изтеглите от Интернет. O'Reilly ни позволи да изпратим тези части (които бяха написани за тази книга) на I&GS, така че те да са от полза на Интернет-базираното Linux общество и да можем да получим коментари и поправки от техните читатели. Благодарим на всички, които помогнаха в редактирането на тези части.

Също така, бихме искали да благодарим на следните хора за тяхната работа върху операционната система Linux - без тях, не би имало за какво да пишем книга ~ Линус Торвалдс, Richard Stallman, Donald Becker, Alan Cox, Remy Card, Eric Raymond, Ted T'so, H.J. Lu, Miguel de Icaza, Ross Biro, Drew Eckhardt, Ed Carp, Eric Youngdale, Fred van Kempen, Steven Tweedie, Patrick Volkerding, Dirk Hohndel, Matthias Ettrichand и всички други хакери, които са твърде много, за да можем да ги изброим тук.

Специални благодарности на следните хора, за техния принос към LDP, техническият преглед на тази книга, приятелство и поддръжка: Phil Hughes, Melinda McBrade, Bill Hahn, Dan Irving, Michael Johnston, Joel Goldberger, Michael K. Johnson, Adam Richter, Roman Yanovsky, Jon Magid, Erik Troan, Lars Wirzenius, Olaf Kirch, Greg Hankins, Alan Sondheim, Jon David, Anna Clark, Adam Goodman, Lee Gomes, Rob Walker, Rob Malda, Jeff Bates и Volker Lendecke. Признателни сме на Shawn Wallace и

Umberto Crenca за великолепната им работа в раздела, посветен на Gimp, в Глава 9.

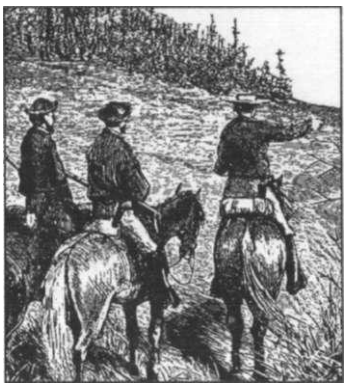
За това трето издание, благодарим на Phil Hughes, Robert J. Chassel, Tony Cappellini, Craig Small, Nat Makarevitch, Chris Davis, Chuck Toporek, Frederic HongFeng и David Pranata за разнообразните коментари и поправки. Особено впечатляващи бяха усилията, вложени от целия екип разработчици и потребители на Debian, организирани от Ossama Othman и Julian T.J. Midgley. Julian организира CVS система за коментари и книгата беше проучена колективно от него, Chris Lawrence, Robert J. Chassell, Kirk Hilliard и Stephen Zander.

Както бихте могли да видите от един бърз поглед върху първите страници на приложенията, получихме голяма помощ от разработчиците, за да документираме поддръжката на Linux и инсталирането му върху различни от Intel системи. Barret G. Lyon и Richard Payne (Alpha), Jason Haas (LinuxPPC), Chris Lawrence (Linux/m68k) и David S. Miller (SPARC) вложиха голяма част от личното си време за да създадат приложенията. Kostas Gewrgiou, Jambi Ganbar, Sanjeev Gupta, Roman Hodek, Geert Uytterhoven, Jes Sorensen, Hubert Figuiere и Christopher F.Miller прегледаха приложенията за различните системи и предложиха корекции.

Ако имате въпроси, коментари или поправки към тази книга, можете свободно да се свържете с авторите. Мат Уелш може да бъде намерен на адрес *mdw@cs.berkeley.edu*, Лар Кауфман на адрес *lark@conserve.org*, а Матиас Далхаймер - на адрес *kalle@dalheimer.de*.

ГЛАВА 1

ВЪВЕДЕНИЕ В LINUX



Тази книга е за Linux - безплатна реализация на Unix за персонални компютърни системи, която поддържа пълна многозадачност, графичната система X Window, TCP/IP мрежа и много други възможности. Концентрирайте се и четете: в страниците, които следват, ще опишем системата детайлно.

Linux създаде повече вълнение в компютърните среди, отколкото която и да е друга разработка в последните години. Неговото изненадващо бързо разпространение и лоялността, която внушава, напомнят за еуфорията от времето на самоделните компютри, характерна за първите години от развитието на компютърните технологии. По ирония, той успява като подмладява една от най-старите операционни системи - Unix. Linux е едновременно нова и стара технология.

В чисто технологични термини, Linux е ядрото на операционната система, което предоставя услуги като многозадачност, виртуална памет, управление на файловата система и устройствата за вход/изход. С други думи, самият Linux е в основата на йерархията на операционната система.

Повечето хора използват термина "Linux" за обозначаване на цялата система - ядрото заедно с многото програми, които се изпълняват върху него - една цялостна среда за създаване на софтуер и друга продуктивна дейност, включваща компилатори, редактори, графични интерфейси, текстообработващи системи, игри и други.

Тази книга ще бъде ваш пътеводител в изменящия се многолик свят на Linux. Linux се разви до операционна система, подходяща за бизнес, обучение и лично творчество, а ние ще ви помогнем да я използвате пълноценно.

Linux може да превърне персоналния компютър в работна станция. Той ще ви даде пълната мощ на Unix. Бизнес-компаниите инсталират Linux на цели мрежи от компютри, използват го за управление на финансова информация и данни в болниците, като среда за разпределена работа, за телекомуникации и т.н. Много университети по света използват Linux за курсове по операционни системи, програмиране и дизайн. И, разбира се, компютърните ентусиасти използват Linux вкъщи за програмиране, създаване на документи и за разнообразни експерименти.

Linux управлява не само работни станции и персонални компютри (за много хора е удобно да използват Linux на преносимите си компютри), но и големи сървъри. Все повече хора откриват, че Linux е мощен, стабилен и достатъчно гъвкав, за да управлява и най-големите дискови масиви и многопроцесорни системи и да изпълнява приложения от Web-сървър до корпоративни бази данни. Учените свързват масиви от Linux машини в огромни групи (клъстери), за да решат проблемите с особено тежки изчисления от областта на физиката и инженерните науки. С последните версии на софтуерния пакет Samba, Linux може да работи дори като Windows сървър за файлове и печат - и то с по-добра производителност от Windows NT!

Това, което прави Linux толкова различен е, че той е *свободна* реализация на Unix. Той е разработван и продължава да се разработва от група доброволци, основно в Интернет, които разменят код, съобщават за открити програмни грешки и решават проблемите в отворена среда. Всеки е добре дошъл да се включи в усилията за разработване на Linux: всичко, което се изисква, е интерес в експериментирането с този безплатен вариант на Unix и известно умение в програмирането.

В тази книга предполагаме, че се справяте с работата с персонален компютър (използващ някаква операционна система, например Windows 95 или някаква версия на Unix). Освен това, предполагаме, че бихте искали да експериментирате, за да накарате всичко да работи правилно - в крайна сметка, това е половината от удоволствието да използвате Linux. Linux се разви до система, която е учудващо лесна за инсталиране и конфигуриране, но понеже е твърде мощна, някои детайли са по-сложни, отколкото в света на Windows. С тази книга като ваш водач се надяваме да откриете, че настройката и използването на ваша собствена Linux система е съвсем лесно и доставя голямо удоволствие.

За книгата

Тази книга е общ поглед и ръководство от начално ниво за Linux. Опитваме се да представим достатъчно обобщена и интересна информация за много теми, за да задоволим еднакво едновременно начинаещите и експертите. Книгата би трябвало да предостави достатъчно материал, така че почти всеки да може да инсталира и използва Linux пълноценно. Вместо да обсъждаме неуловимите технически детайли - неща, които се променят при бързата разработка на Linux - ние ви даваме достатъчно основни знания, за да откриете останалото самостоятелно.

Книгата е предназначена за хората, които наистина искат да използват мощта, предоставяна от Linux. Вместо да тълкуваме сложните детайли, тук ви даваме достатъчно информация, за да разберете правилно как работят различните части от операционната система, така че да можете да ги изменяте по желан от вас начин, да ги настройвате и сами да решавате възникналите проблеми със системата.

Linux не е труден за инсталиране и употреба. Но както с всяка друга реализация на Unix, понякога е нужна малко черна магия, за да настроите всички детайли коректно. Надяваме се, че тази книга ще ви качи на екскурзионния автобус на Linux и ще ви покаже колко добра може да бъде тази операционна система.

В книгата ще разгледаме следните въпроси:

- Какво е Linux? Дизайнът и философията на тази уникална операционна система и какво може да направи тя за вас.
- Всички детайли, от които се нуждаете, за да използвате Linux, включително препоръки за това, какъв тип хардуерна конфигурация е подходящ за една завършена система.
- Как да получите и инсталирате Linux. Обсъждаме дистрибуциите на Red Hat, SuSE и Debian по-подробно от останалите, но основните принципи са приложими за всяка версия на Linux.
- За нови потребители предлагаме въведение в Unix, включително общи сведения за най-важните команди и концепции.
- Грижата за Linux и неговата поддръжка, включително системна администрация и поддръжка, актуализиране на системата и отстраняване на възникнали проблеми.
- Извличане на максимума от вашата Linux система, с мощни инструменти като TEX, Emacs, KDE и други.

1 \ • Linux като среда за програмиране. Инструментите, с които се програмира и разработва софтуер в Linux. Запознаваме ви с компилирането и отстраняването на грешки в програми на C и C++, Java, Perl, скриптове за командния интерпретатор и Tcl/Tk.

• Използване на Linux за телекомуникация и мрежова работа, включително основите на TCP/IP, PPP за достъп до Интернет чрез модем, конфигуриране на ISDN, електронна поща и достъп до Web. Ще ви покажем дори как да конфигурирате Linux като web-сървър.

Има милиони неща, които с удоволствие бихме ви казали за Linux. За съжаление, ако обсъдим всичките, книгата би била с размера на пълния *Oxford English Dictionary* и би било невъзможно за когото и да било (без да споменаваме горките автори) да я поддържа. Вместо това се опитваме да включим най-важните и интересни страни на системата и да ви покажем как да научите повече.

Глава 4
Глава 5

Макарче голяма част от дискусиите в книгата не са прекалено технически, би ви помогнало, ако имате опит с друга Unix система. За тези, които нямат опит, сме включили кратко ръководство в Глава 4, *Основни команди и концепции в Unix.* Глава 5, *Основи на системната администрация,* описва изцяло администрирането на системата и може да бъде от полза дори на опитни потребители на Unix, които използват Linux.

Прилож.А

Ако нямате опит с Unix, ще ви е полезно да прочетете едно по-обширно ръководство за него. Тук не се спираме на основите задълго; вместо това предпочитаме да преминем към по-приятната част на системата. Макар че в началото тази книга би трябвало да е достатъчна, за повечето читатели ще е полезно да научат повече за използването на Unix и многобройните му инструменти. В приложение A, *Източници на информация за Linux,* можете да намерите списък с подходяща литература.

Кратка история на Linux

Unix е една от най-популярните операционни системи по света, поради голямата си поддръжка и разпространение. Тя е разработена в средата на 70-те години като многозадачна операционна система за миникомпютри и големи машини. След това Unix се разраства и днес е една от най-широко използваните операционни системи по света, независимо от понякога объркващия й интерфейс и липсата на стандартизация.

Каква е истинската причина за популярността на Unix? Много хакери смятат, че Unix е Истинската Операционна Система. Именно затова Li-

nux се разработва от разрастваща се група Unix хакери, които искат да създадат собствена операционна система.

Съществуват версии на Unix за много системи - от персонални компютри до супер-компютри като Cray Y-MP. Повечето версии на Unix за персонални компютри са доста скъпи и неудобни. В момента лиценз за един компютър на Unix Version V на AT&T за процесори, съвместими с Intel 386, струва около 1 500 щатски долара.

Linux е безплатно разпространявана версия на Unix, първоначално разработена от Линус Торвалдс, който започва работата върху нея през 1991 година като студент в Университета в Хелзинки, Финландия. В момента Линус работи за Transmeta Corporation в Санта Клара, Калифорния и продължава да поддържа *ядрото* на Linux, т.е. сърцевината на операционната система.

Линус разпространява началната версия на Linux безплатно по Интернет и неволно поражда един от най-големите феномени в разработката на софтуер. Днес Linux се пише и поддържа от група от няколко хиляди (ако не и повече) разработчици, които си сътрудничат чрез Интернет. Появиха се компании, които предлагат поддръжка за Linux, създават лесни за използване дистрибуции и продават работни станции с предварително инсталиран Linux. През март 1999 година се проведе първото търговско изложение Linux World Expo в Сан Хосе, Калифорния, което беше посетено от около 12 000 души. Предполага се, че броят на потребители на Linux е около 10 милиона (а ние очакваме, че това да няма да ви изглежда много по времето, в което четете тази книга).

Вдъхновен от операционната система Minix на Andrew Tanenbaum (друга безплатна реализация на Unix за PC-та, макар и доста опростена), Linux започва като училищен проект на Линус, който е искал да създаде проста Unix система, способна да работи на персонален компютър с процесор Intel 386. Първото споменаване на Linux е в Usenet в групата *comp.os.minix.* В началото това е била дискусия най-вече за разработката на малка, академична Unix система за потребители на Minix, които искат повече възможности.

Най-ранната разработка на Linux се отнася предимно до способностите за многозадачност в защитения режим на процесора 80386 и е написана изцяло на асемблер. Линус пише:

> "След това беше лесно: кодът все още беше недодялан, но аз имах някои (полезни) устройства и изчистването на грешки беше по-лесно. На този етап започнах да използвам C и това чувствително ускори разработката. По това време се замислих сериозно за мегаломанските си идеи да направя

"по-добър Minix от Minix". Надявах се, че някой ден ще мога да компилирам *gcc* под Linux ...

Минаха два месеца от началото на проекта, но малко след това имах драйвер за твърдия диск (с доста грешки, обаче работеше на моята машина) и малка файлова система. По това време разпространих версия 0.01 (късния август на 1991): тя не беше добра, нямаше драйвер за дискетно устройство и не можеше почти нищо. Не мисля, че някой дори е компилирал тази версия. Но аз бях вманиачен и не исках да спирам, докато не успея да изхвърля Minix."

За Linux, версия 0.01, няма коментари. Изходният код за тази версия дори не е изпълним: той съдържа само скелета на ядрото и се предполага, че имате достъп до Minix, за да го компилирате и изпробвате.

На 5 октомври 1991 година Линус съобщава за първата "официална" версия на Linux - версия 0.02. На този етап Линус може да стартира *bash* (GNU Bourne Again Shell - командният интерпретатор от проекта GNU) и *gcc* (C/C++ компилаторът на GNU), но почти нищо друго не работи. Това отново е само една хакерска система. Основното в разработката е било създаването на ядрото. На поддръжката на потребителите, документацията, разпространението и т.н. изобщо не се обръща внимание. Днес ситуацията е съвсем различна - истинското удоволствие в света на Linux е свързано с графичните потребителски среди, лесните за инсталиране дистрибуции и приложенията от високо ниво, като графични и други продуктивни програми.

Линус пише в *comp.os.minix:*

"Тъгувате ли за хубавите дни на Minix-1.1, когато мъжете бяха истински мъже и си пишеха собствени драйвери? Имате ли хубав проект или просто се уморявате до смърт, докато се опитвате да настроите операционната система за вашите нужди? Смятате ли че е разочароващо, че всичко работи добре под Minix? Без да прекарвате безсънни нощи, за да накарате новите програми да работят? Ако е така, може би това писмо е за вас.

Както отбелязах преди един месец, аз работя върху безплатна версия на подобна на Minix операционна система за компютри AT-386. Накрая тя стигна до етап, в който може дори да се използва (макар че може и да не е - зависи от това, какво искате) и бих искал да предоставя изходния код за по-широко разпространение. Тя е само версия 0.02 ... но успявам да стартирам *bash, gcc,* GNU *make,* GNU *sed, compress* и т.н. под нея."

След версия 0.03 Линус прескача на версия 0.10, тъй като повече хора започват да работят върху Linux. След още няколко преработки Линус увеличава номера на версията до 0.95, за да отрази очакванията, че системата скоро ще бъде готова за "официално" представяне. (Обикновено на софтуера не се дава версия 1.0, докато не бъде завършен и без грешки

поне теоретично.) Това става през март 1992 година. Почти година и половина по-късно, декември 1993, ядрото на Linux беше все още версия 0.99.pll4, доближавайки се до 1.0. Версия 1.0 се появява през март 1994. В моментаДмарт 2000 година) текущата версия на ядрото е 2.2.14, а версиите 2.3 се разработват. (По късно ще обясним подробно конвенциите при номерирането на версиите на Linux.)

Linux не би съществувал без програмите от проекта GNU, създадени от Free Software Foundation (Фондация за свободен софтуер). Компилаторът *gcc* от този проект (виж Глава 13, *Езици за програмиране)* даде живот на кода на Линус Торвалдс. Софтуерът на GNU се използва за разработката на Linux от самото начало. Заради изключително важния му принос Free Software Foundation дори настоява дистрибуциите на Linux с програмите на GNU да се наричат GNU/Linux.

Глава 13

Разработката на Университета в Бъркли - BSD Unix (Berkeley Software Distribution) също има важно значение за Linux, но не толкова за създаването му, колкото чрез предоставянето на програми, които го правят популярен. Повечето основни програми в дистрибуциите на Linux са адаптирани от BSD. Особено важни са мрежовите инструменти и т.нар. "демони" (daemons). Кодът в ядрото, реализиращ поддръжката на мрежа, е независима разработка (всъщност, разработван е два или три пъти), но мрежовите инструменти и демони са взети от BSD.

Днес Linux е завършен вариант на Unix, способен да използва системата X Window, TCP/IP, Emacs, Web, email, news и т.н. Почти всеки важен безплатен софтуерен пакет е адаптиран за Linux, а вече съществува и немалко комерсиален софтуер за него. Всъщност, много разработчици създават приложенията си за Linux, а след това ги адаптират за други Unix системи. Днес ядрото на Linux поддържа много повече хардуер от оригиналните си версии. Много хора са провели тестове за производителността на системи с Linux и са открили, че тя е по-голяма от производителността на работни станции на Sun Microsystems и Compaq. Linux се представя по-добре от (или поне равностойно на) Windows 98 и Windows NT на разнообразни тестове. Едва ли някой е предполагал, че този "малък" вариант на Unix ще порасне дотолкова, че да превземе целия свят на персоналните компютри и сървъри.

Все повече Unix програмисти започват да използват Linux заради ниската му цена - само за няколко долара те получават цялостна среда за програмиране, при това работеща и на евтини компютри и разполагаща с огромно количество адаптирани програми. Linux е модерна операционна система, съвместима със стандарта POSIX и изглеждаща почти като System V, затова кода, който работи под Linux, би трябвало да работи под която и да е друга съвременна Unix операционна система. Дори на скромни PC-та Linux работи по-бързо от много Unix работни станции.

Работата в мрежа е една от силните страни на Linux. Той е изключително подходящ за мрежи като Free-Nets, мрежи на организации с идеална цел или свободни потребителски общества, които използват протокола UUCP. Linux поддържа Network File System (NFS) и Network Information Service (NIS), което улеснява свързването на персонален компютър към корпоративна или академична мрежа с други Unix машини. С Linux лесно можете да използвате файлове съвместно с други машини, да влезете в отдалечена машина (или да позволите на отдалечени потребители да влязат във вашата машина) и да стартирате приложение на нея. Linux поддържа също и софтуерния пакет Samba, което му дава възможност да работи като Windows сървър за файлове и печат, при това според много хора по-бързо (и по-евтино) от Windows NT.

Хакерите на ядрото бяха първите, които започнаха да използват Linux. Всъщност, те са тези, които помогнаха на Линус Торвалдс да създаде Linux и все още са внушително общество. Ако искате да настроите размера на буферите или броя на елементите в таблица, за да ускорите малко работата на вашето приложение, Linux е една от най-добрите алтернативи. А ако възникнат проблеми, ще намерите много отзивчиви хора в Мрежата.

Linux се превръща в чудесен форум за мултимедия. Това е резултат от съвместимостта му с огромно разнообразие от хардуер, включително основните модерни звукови и видео контролери. За Linux са адаптирани различни среди за програмиране, включително MESA 3D toolkit (безплатна реализация на OpenGL). Приложението GIMP (безплатна програма, подобна на Adobe Photoshop) е разработено под Linux и много художници го използват за обработка на графика и дизайн. Например, при създаването на филма "Титаник", някои от специалните ефекти са направени с Linux машини с процесори Alpha.

Linux има и други практически приложения. Linux системи са преминали през откритията океан в северния Атлантик, управлявайки телекомуникациите и анализа на данни в океанографски изследователски кораб. Linux системи се използват в изследователските станции в Антарктика.

Linux се използва в много болници за поддръжка на информацията за пациентите им. Един от рецензентите на тази книга използва Linux в Американските Военноморски Сили. Linux доказва, че е стабилен и използваем като останалите реализации на Unix.

И така Linux се разпространява в най-различни посоки. Могат да му се наслаждават дори наивните крайни потребители, ако получат поддръжката, която университетите и корпорациите обикновено оказват на своите компютърни потребители. Настройването и поддръжката му изискват определени умения, но Linux доказа, че е мощна операционна система с добра цена, коятопредоставя на нови възможности на всеки, който иска да има допълнителен контрол върху компютъра си.

Възможности на Linux

Linux има повечето от възможностите на другите реализации на Unix, а освен това има много възможности, които не се срещат другаде. В този раздел ще направим бърз преглед на възможностите на ядрото.

Пояснение на начина на номериране на версиите на Linux

За някои от начинаещите потребители на Linux е объркващ начинът, по който се отбелязва версията на различните части на софтуера. Когато за пръв път видите Linux, вероятно това ще бъде дистрибуция на компакт-диск с надпис "Red Hat Linux версия 5.2" или "SuSE Linux версия 6.0". Важно е да разберете, че номерът на версията се отнася само за конкретната дистрибуция (която представлява Linux в комплект с голямо количество безплатни пакети с приложения и обикновено се продава на компакт-диск). Този номер се определя от Red Hat, SuSE или Debian и затова може да няма нищо общо с номера на версията на софтуера в тази дистрибуция. Не се подлъгвайте - това, че една компания използва по-голям номер на версия за дистрибуцията си, не означава, че софтуерът в нея е по-съвременен.

Ядрото на Linux, както и всяко приложение, компонент, библиотека или софтуерен пакет в дадена дистрибуция на Linux има *собствен* номер на версия. Например, можете да използвате *gcc* версия 2.7.2.3 едновременно с графичния интерфейс XFree86 с версия 3.3.1. Както се досещате, колкото по-голям е номерът на версията, толкова по-нов е софтуерът. Когато инсталирате дистрибуция (например Red Hat или SuSE), всичко

това се опростява, защото обикновено в дистрибуциите се включват най-новите версии на всеки пакет.

Ядрото на Linux получава номера на версията си по специален начин, който трябва да разбирате. Както отбелязахме по-рано, ядрото е сърцето на операционната система и отговаря за управлението на всички хардуерни ресурси във вашия компютър - твърд диск, мрежови платки, памет и т.н. За разлика от Windows, ядрото не включва библиотеки от приложно ниво или графичен потребителски интерфейс. В известен смисъл като потребител никога няма да комуникирате с ядрото директно - обикновено вместо това се използва интерфейс като команден интерпретатор или графичната среда (ще обсъдим това по-късно).

Все още много хора смятат версията на ядрото на Linux за версия на "цялата система", което е заблуждаващо. Ако чуете някой да казва, че "използва ядро с версия 2.3.32", това не означава много, ако целия останал софтуер в системата е остарял.

Ядрото на Linux получава номера на версията си по следния начин. Винаги са на разположение (тоест могат да се изтеглят от Интернет) *две* "последни" версии на ядрото - "стабилна" и "разработвана". Стабилната версия е предназначена за повечето потребители на Linux, които не се интересуват от разработката на несигурни, експериментални възможности на ядрото, а се нуждаят от стабилна, работеща система. От друга страна, разработваната версия се променя доста често, като разработчиците добавят и изпробват нови възможности. Промените в стабилната версия се правят предимно за отстраняване на открити грешки или пропуски в сигурността на системата, докато промените в разработваната версия могат да са от всякакъв тип - от добавяне на нова подсистема към ядрото до фини настройки на някой драйвер за подобряване на производителността му. Разработчиците на Linux не гарантират, че разработваната версия на ядрото ще работи на всеки компютър, но се грижат стабилната версия да работи добре навсякъде.

Стабилната версия на ядрото има четен второстепенен номер (например 2.2), докато разработваната версия има нечетен (например 2.3). Текущата разработвана версия винаги има второстепенен номер точно с едно по-голям от второстепенния номер на текущата стабилната версия. Така че, когато текущата стабилна версия стане 2.4, текущата разработвана версия ще бъде 2.5. (Разбира се, ако Линус реши да преименува версия 2.4 на 3.0, разработваната версия ще стане 3.1).

Всяка от версиите на ядрото има асоциирано трето число - номер на поправката (patch, или "кръпка"), например 2.2.19 или 2.3.85. Номерът на поправката показва конкретна преработка на ядрото от дадена версия

като по-големите числа съответстват на по-нови поправки. В момента (март 2000), последната стабилна версия е 2.2.14, а последната разработвана - 2.3.43.

Море от възможности

Linux е истинска многозадачна и многопотребителска операционна система (точно като всяка друга версия на Unix). Това означава, че много потребители могат едновременно да влязат в една машина, като всеки от тях може едновременно да използва много програми. Освен това, Linux поддържа многопроцесорни платформи (като компютрите, имащи дънни платки с два процесора Pentium). В момента се поддържат до 16 процесора на платформа, което е много полезно за високопроизводителни сървъри и за някои научни приложения.

Linux е съвместим с много Unix стандарти (доколкото в Unix има стандарти) на ниво изходен код, включително IEEE POSIX.l, System V и BSD. При разработването на Linux една от целите е била изходния код да бъде преносим, затова вероятно ще откриете, че част от възможностите на Linux се използват и в други реализации на Unix. Голяма част от безплатния софтуер за Unix, който можете да намерите в Интернет, се компилира на Linux без проблеми.

Ако имате познания за Unix, може би ще ви заинтересуват някои специфични вътрешни възможности на Linux, между които управлението на задачи по стандарта POSIX (използва се от командни интерпретатори като *tcsh* и *bash),* псевдо-терминалите (устройствата *pty)* и поддръжката на различни клавиатурни подредби чрез използване на динамично зареждаеми клавиатурни драйвери. Linux поддържа и *виртуални конзоли,* което ви позволява да превключвате между много потребителски сесии от системната конзола в текстов режим. Потребителите на програмата *screen* ще установят, че реализацията на виртуални конзоли в Linux им е позната.

Linux може безпроблемно да съществува едновременно с друга операционна система върху една машина, например с Windows 95/98, Windows NT, OS/2 или други версии на Unix. Програмата за зареждане на Linux LILO (LInux LOader) ви позволява да изберете коя операционна система да се зареди по време на стартирането на компютъра, а Linux е съвместим с други аналогични програми (например с програмата за зареждане на Windows NT).

Linux може да работи с широка гама процесорни архитектури, включително Intel x86 (386, 486, Pentium, Pentium Pro, II и III), SPARC, Alpha,

PowerPC, MIPS и m68k. Разработват се адаптации за различни други архитектури и се очаква, че Linux ще работи прекрасно на следващото поколение процесори на Intel - чиповете Merced. Разработен е дори вариант на Linux за вградени процесори като чипа в електронния помощник PalmPilot на 3Com.

Linux поддържа разнообразни типове файлови системи за съхраняване на данни. Някои файлови системи са разработени специално за Linux, например Second Extended FileSystem *{ext2fs* - втора разширена файлова система). Поддържат се и други типове файлови системи, например Minix-1 и Xenix. Реализирана е поддръжка и на файловата система MS-DOS, което дава на Linux директен достъп до файлове на Windows и MS-DOS, записани върху твърд диск или дискета. Поддържат се и файловите системи на OS/2, Apple, Amiga и Windows NT, а също и файловата система ISO 9660, с която могат да се четат всички стандартни формати компакт-дискове. Ще поговорим повече за файловите системи в Глава 3, *Инсталиране и начално конфигуриране,* и в Глава 5.

Глава 3
Глава 5

Поддръжката на работа в мрежа е една от най-силните страни на Linux, едновременно като функционалност и като производителност. Linux предоставя пълна реализация на TCP/IP мрежа. Това включва драйвери за много от разпространените Ethernet карти, реализация на протоколите PPP и SLIP (които ви дават достъп до TCP/IP мрежа през серийна връзка), Parallel Line Internet Protocol (PLIP) и мрежовата файлова система NFS. Linux поддържа пълен спектър клиенти и услуги за TCP/IP, например FTP, Telnet, NNTP и протокола за изпращане на e-mail SMTP. Ядрото на Linux включва цялостна поддръжка за мрежов firewall (това е система, която наблюдава пакетите по мрежата, като предотвратява например непозволен достъп до вътрешна мрежа). Всяка Linux машина може да бъде конфигурирана като firewall. Широко известно е, че мрежовата производителност под Linux е по-добра от тази на много други операционни системи. Поддръжката на мрежа е разгледана по-подробно в Глава 15, *TCP/IPuPPP.*

Ядрото

Ядрото е същността на операционната система. То е кода, който управлява взаимодействието между програмите и хардуерните устройства, реализира многозадачността и контролира много други свойства на системата. Ядрото не е отделен процес, който работи в системата. Можете да мислите за него по-скоро като за множество програми, които непрекъснато се намират в паметта и всеки процес има достъп до тях. Програмите от ядрото могат да се използват по много начини. Пример за ди-

ректно използване на ядрото от процес в компютъра е изпълняването на системна заявка. Като резултат ядрото извършва определени действия, полезни за извикващия процес. Например, системната функция *read* ще прочете данни от зададен файл. За програмистите това изглежда като още една функция на C, но всъщност кодът, който изпълнява *read,* се намира в ядрото.

Код от ядрото се изпълнява и в други случаи. Например, когато хардуерно устройство генерира прекъсване, кодът за обработката на прекъсването се намира в ядрото. Когато някой процес извърши действие, при което е необходимо да изчака резултат, ядрото спира този процес и пуска друг вместо него. По подобен начин ядрото бързо спира или пуска процеси като използвайки прекъсването, генерирано от часовника на компютъра, (а също и други начини) за промяна на активния процес. Общо взето, това е начинът, по който се постига многозадачност.

Ядрото на Linux е известно като *монолитно,* което означава, че всички драйвери за хардуерни устройства са част от него. Някои операционни системи използват архитектура, известна като *микро-ядро,* при която драйверите за устройствата и други компоненти (като файловите системи и управлението на паметта) *не са* част от ядрото. Тези модули се третират подобно на обикновените потребителски програми. И двете архитектури имат предимства и недостатъци. Монолитната архитектура е поразпространена в реализациите на Unix и се използва от класическите ядра, например в System V и BSD. Linux поддържа зареждаеми драйвери (които могат да се заредят и отстранят от паметта с потребителски команди); това е темата на раздела "Зареждаеми драйвери" в Глава 7, *Актуализиране на софтуера и ядрото.*

В много архитектури ядрото на Linux може да емулира инструкции за работа с плаваща запетая, така че системите без математически копроцесор могат да изпълняват програми, които изискват такъв копроцесор.

Ядрото на Linux за платформи с процесор Intel е разработено така, че да използва специалните възможности на защитения режим на фамилията x86 (започвайки от i386). По-конкретно Linux използва управлението на паметта в защитен режим, базирано на дескриптори и още много от съвременните възможностите на тази фамилия. Всеки запознат с програмирането в защитен режим на 80386 знае, че този чип е създаден за многозадачни операционни системи като Unix и Linux използва тази функционалност.

Ядрото на Linux поддържа "зареждане при необходимост". Това означава, че в паметта на компютъра се зареждат само тези части от програмата, които в действителност се използват. Освен това, ако работят еднов-

ременно няколко екземпляра на програмата, в паметта се намира само едно копие от кода й.

За да увеличи количеството налична памет, Linux поддържа виртуална памет. Това означава, че част от твърдия диск се използва за "уголемяване" на паметта на компютъра. Тази част се нарича *място за swap.* Когато системата се нуждае от повече физическа памет, тя записва част от неактивните страници на диска и използва освободената памет за други цели. Това позволява на системата да стартира по-големи приложения и едновременно да поддържа повече потребители. Разбира се, виртуалната памет не е заместител на физическата памет; тя е много по-бавна поради голямото време, което е нужно за достъп до диска.

Освен това ядрото реализира унифициран пул на паметта за потребителските програми и дисковия кеш.[1] По този начин цялата свободна памет се използва за кеш. Ако се изпълнява програма, изискваща повече памет, размерът на кеша се намалява.

Linux поддържа динамично свързани споделени библиотеки. Това означава, че програмите използват общ код, който се намира в единствен файл (библиотека) на диска, за разлика от механизма на SunOS за споделени библиотеки. По този начин изпълнимите файлове заемат значително по-малко място на диска, особено тези от тях, които използват много функции от библиотеките. Освен това, в определен момент в паметта се намира само едно копие от кода в библиотеките, което чувствително намалява количеството на необходимата памет. Съществуват и статично свързани библиотеки за тези, които искат да имат "цялостни" изпълними програми, които работят без да е необходимо да има инсталирани споделени библиотеки. Понеже споделените библиотеки в Linux се свързват динамично при пускането на програма, програмистите могат да заменят модули от библиотеките със свои собствени.

За да улесни отстраняването на грешки, при срив по време на работата на дадена програма, ядрото на Linux записва във файл копие от паметта на програмата по време на грешката (този процес се нарича core dump).

* Т.е. място за замяна. Това име не е съвсем точно - понякога на диска се прехвърля не целият процес, а само отделни страници от паметта, която той заема. Разбира се, има много случаи, в които на диска се прехвърля целият процес, но това не е задължително.

' Метод, при който данните, прочетени от твърдия диск, се съхраняват в паметта, така че при повторна необходимост се четат от паметта, а не от значително по-бавния диск. Така се постига ускоряване на работата. - б.пр.

Като се използва този файл и програмата, компилирана с подходящи опции, може да се установи каква е причината за проблема. Ще обсъдим това в раздела "Анализ на core dump" в Глава 14, *Инструменти за програмисти.*

Софтуер

В този раздел ще ви представим много софтуерни приложения за Linux и ще поговорим за основните задачи при работа с компютър. В крайна сметка, най-важната част от системата е широката гама софтуер за нея. Още по-впечатляващо е, че повечето програми за Linux се разпространяват свободно.

Основни команди и полезни програми

Практически целият стандартен софтуер за Unix е адаптиран и за Linux. Това включва основни команди като *ls, awk, tr, sed, bc, more* и т.н. Съществуват адаптации за Linux на повечето популярни софтуерни пакети, включително Perl, Phyton и Java Development Kit. Затова при работа с Linux можете да използвате вашата позната работна среда от други Unix системи. Имате всички стандартни команди и полезни програми. (Потребителите без опит с Unix трябва да прочетат Глава 4, за да се запознаят с основните команди на Unix.)

Linux предоставя много текстови редактори, включително ***vi*** (както и модерните му версии, например *vim), ex, pico* и *jove,* както и GNU Emacs и негови варианти - XEmacs (който включва разширени възможности при работа под системата X Window) и *joe.* Почти сигурно е, че ще намерите адаптация за Linux на текстовия редактор, с който сте свикнали.

Изборът на текстов редактор е интересен проблем. Много потребители на Unix все още използват "прости" редактори като ***vi*** (всъщност, за да напишат тази книга, авторите използваха vi и Emacs под Linux). Все пак ***vi*** има много ограничения, тъй като е много стар текстов редактор. Затова все повече нараства популярността на по-модерните (и сложни) редактори като Emacs. Emacs поддържа пълен, базиран на LISP макро-език и интерпретатор, мощен команден синтаксис и други полезни възможности. Съществуват пакети с макроси за Emacs, с които можете да четете електронна поща и новини в Usenet, да редактирате съдържанието на директории и дори да участвате в психо-терапевтичен сеанс, воден от изкуствен интелект (незаменим за изнервените хакери на Linux). В Глава 9, *Редактори, текстови инструменти, графика и печат,* сме включили пълно ръководство за vi и описваме детайлно Emacs.

Интересно е да се отбележи, че повечето от основните програми в Linux са от проекта GNU. Софтуерът на GNU има усъвършенствани възможности, които не се срещат в стандартните версии на аналогичните програми от BSD или от AT&T. Например, GNU-версията на текстовия редактор *vi,* наречена *elvis,* включва структуриран макро-език, който се отличава от оригиналната версия на AT&T. Все пак, стремежът в проекта GNU е да се запази съвместимостта на софтуера с оригиналите от BSD и System V. Много хора смятат, че GNU-версиите на тези програми са подобри от оригиналите. Пример за това са програмите за компресиране *gzip* и *bzip2,* които архивират данни с много по-голяма ефективност от оригиналната програма за Unix - *compress.*

Най-важната програма за много потребители е *командния интерпретатор* (shell). Това е програма, която чете и изпълнява командите на потребителя. Освен това, много от командните интерпретатори предоставят възможности като *управление на задачите* (job control - това позволява на потребителя да управлява няколко едновременно работещи програми и не е толкова сложно, колкото звучи), пренасочване на входния и изходния поток и команден език, с който могат да се пишат *скриптове за командния интерпретатор.* Скрипт за командния интерпретатор е всеки файл, съдържащ програма на езика на командния интерпретатор, аналогично на "batch файловете" в MS-DOS.

Съществуват много видове командни интерпретатори за Linux. Най-важната разлика между тях е командният им език. Например, C shell *(csh)* използва команден език, подобен на езика за програмиране C. Класическият Bourne Again shell използва друг език. Изборът на команден интерпретатор често зависи от езика, който той предоставя. Използваният от вас команден интерпретатор в известна степен определя работната ви среда под Linux.

Независимо с кой команден интерпретатор за Unix сте свикнали да работите, най-вероятно ще намерите негова версия, адаптирана за Linux. Най-популярен е GNU Bourne Again Shell *{bash),* който е вариант на Bourne shell, *bash* има много подобрения като управление на задачите, списък на използваните команди, автоматично допълване на командите и имената на файловете, подобен на Emacs (или при ваше желание - на vi) интерфейс за редактиране на командния ред и мощни разширения към стандартния език на Bourne shell. Друг популярен команден интерпретатор е *tcsh,* който е версия на C shell с подобрена функционалност, подобна на тази в *bash.* Предлагат се още Korn shell *(ksh),* интерпретатора на BSD *ash, zsh* (малък команден интерпретатор, подобен на Bourne shell) и *rc* (командният интерпретатор от Plan 9).

Защо обръщаме толкова внимание на тези основни програми? Linux ви дава уникалната възможност да настроите системата според нуждите си. Например, ако вие сте единственият човек, който използва системата и предпочитате да използвате единствено редактора ***vi*** и командния интерпретатор *bash,* няма причина да инсталирате друг редактор или команден интерпретатор. Много потребители и хакери на Linux предпочитат сами да направят системата си.

Текстообработка и създаване на документи

Почти всеки компютърен потребител се нуждае от някакъв тип система за създаване на документи (колко компютърни ентусиасти познавате, които все още използват молив и лист? Обзалагаме се, че не са много). В света на персоналните компютри е задължително да има възможност за създаване на документи (така наречения *word processing).* Това включва редактиране и обработка на текст и отпечатването му върху хартия. Често за тази цел се използват приложения от тип WYSIWYG (What-You-See-Is-What-You-Get - това, което виждате, е това което получавате). Обикновено текстовите документи съдържат различни фигури, таблици и други елементи.

В света на Unix *текстообработката* е много по-обширно понятие и е доста различна от класическата концепция за създаване на документи (word processing). Когато използва текстообработваща система, авторът описва форматирането на текста със специален "полиграфски език". Вместо да използва специално приложение за създаване на документи, авторът може да въвежда и редактира текста с произволен текстов редактор като *vi* или Emacs. След като завърши въвеждането на изходния текст (използвайки полиграфския език), авторът го форматира с отделна програма, която превръща текста в подходящ за отпечатване формат. Това е донякъде аналогично на програмирането на език като C и "компилирането" на документа за печат.

Съществуват много текстообработващи системи за Linux. Една от тях е *grojf* GNU-версията на *troff* - класическа програма за форматиране на текст, разработена от Bell Labs, която все още се използва в много Unix системи по света. Друга модерна система за текстообработка е TfiX, създадена от Доналд Кнут, известен с разработките си в областта на компютърните науки. Освен това в Linux можете да използвате диалекти на TEX - например LATEX.

Текстовите процесори като TEX и *groff* се различават предимно в синтаксиса на полиграфския език, който използват. Изборът на текстообра-

ботваща система зависи и от предлаганите помощни програми за нея, както и от личния вкус на автора.

Например, някои хора смятат, че полиграфския език на *groff* е малко неясен и затова използват TEX, който е по-лесно разбираем. Все пак, *grojf* може да създаде обикновен ASCII файл, който може да се разглежда на терминал, докато системата TEX е предназначена преди всичко за създаване на документи за печат. Разбира се, съществуват различни програми, с които можете да създадете обикновен ASCII файл от форматиран за TEX документ или да превърнете документ от TEX в groff формат.

Друга текстообработваща система е Texinfo. Тя е разширение на TEX и се използва от Free Software Foundation за документиране на софтуер. От един и същ изходен файл Texinfo може да създаде документ за печат и електронен хипертекстов документ във формат "Info". Info-файловете са основния формат за документация, който се използва от GNU софтуера (например от Emacs).

Текстовите процесори се използват широко от компютърното общество за създаване на вестници, дисертации, статии за списания и книги (тази книга е написана във формат LATEX, филтрирана в създадена от нас SGML система и е отпечатана от *groff).* Възможността да се обработи изходния документ като обикновен текстов файл облекчава създаването на много разширения на самия текстов процесор. Изходният документ не е записан в неясен формат, който може да се чете само от конкретно приложение (word processor), затова програмистите могат да напишат програми, анализиращи или превеждащи езика в друг формат и по този начин да разширят системата.

Как изглежда един език за форматиране? Най-общо, изходният файл се състои преди всичко от самия текст и от "команди", с които се създава някакъв конкретен ефект, например смяна на шрифт, установяване на размера на страницата, създаване на списъци и т.н.

Най-известният език за форматиране на текст е HTML. XML е негов помлад братовчед. HTML произлиза от SGML (тази книга се поддържа във формат SGML). Конкретните маркери (tags), които се използват в книгата (като <PARA> и <TITLE>), са описани в Docbook - стандартно множество от маркери, което се използва за създаване на техническа документация, например на официалната документация на Linux (виж подолу в тази глава). Следва едно типично начало на раздел според стандарта в Docbook:

```
<sect2xtitle>0cHOBHM команди и полезни nporpaMM</title>

<para>

Практически целият стандартен софтуер за Unix е адаптиран и за
Linux. Това включва основни команди като <command>ls</command>,
<command>awk</command>, <command>tr</command>,
<command>sed</command>, <command>bc</command>,
<command>more</command> и т.н.  Съществуват адаптации за Linux на
повечето популярни софтуерни пакети, включително Perl, Phyton и
Java Development Kit. Затова при работа с Linux можете да използвате
вашата позната работна среда от други Unix системи. Имате всички
стандартни команди и полезни програми.  (Потребителите без опит с
Unix трябва да прочетат <xref linked="X-100-3-chap-basic">, за да се
запознаят с основните команди на Unix.)

</para>
```

На пръв поглед полиграфският език изглежда малко неясен, но всъщност е доста лесен за изучаване. Използването на текстообработваща система налага спазването на полиграфски стандарти при писане. Например, всички номерирани списъци в документа ще изглеждат по един и същ начин, докато авторът не промени полиграфската дефиниция за "номериран списък".

Основната цел на полиграфските езици е да позволят на автора да се концентрира върху писането на самия текст, вместо да се грижи за полиграфските проблеми. Когато горният пример се отпечатва, издателят може да избере какви да бъдат шрифта, цвета и другите параметри на командите между маркерите **<command>** и **</command>.** Лесно е да се създаде индекс със страниците, на които се срещат тези команди. Освен това, точният номер и име на глава ще бъдат сложени на местата, където се среща изглеждащия странно маркер **<xref** &>. По този начин текстът е коректен, дори ако авторът пренареди главите, след като напише конкретния параграф.

Макар че има редактори за HTML (и дори няколко за SGML) от тип WYSIWYG, умението да се пишат "ръчно" маркери като използваните в предишния пример, изисква съвсем малко практика. След това със съответните инструменти документа може да се "преведе" в стандартен формат (като PostScript или PDF) и да се разгледа на екрана или да се отпечата на принтер. В Глава 9 се разглеждат текстообработващи системи като LATEX, Texinfo и *troff*.

Приложения от тип WYSIWYG са привлекателни по много причини. С тях може да се създаде мощен, а понякога и сложен визуален интерфейс за редактиране на документи. Все пак, този интерфейс има ограничени възможности за художествено оформление. Затова много от тези приложения предоставят специален "език за форматиране" за въвеждане на

сложни изрази, например на математически формули. Този език е подобен на полиграфския език, но е с по-малко възможности.

Основната полза от полиграфския език на текстообработващите системи е възможността да укажете точно какво искате. Освен това, текстообработващите системи ви позволяват да редактирате изходния текст с произволен текстов редактор и лесно да го превърнете в друг формат. Недостатъкът на тази гъвкавост и мощ е липсата на WYSIWYG интерфейс.

Много потребители на WYSIWYG приложения за създаване на документи са свикнали да виждат форматирания текст докато го редактират. От друга страна, при използване на текстов процесор, авторът обикновено не се грижи за това, как ще изглежда текстът след като се форматира. Авторът се научава да определя как изглежда документа, използвайки командите за форматиране в изходния текст.

Съществуват програми, които ви позволяват да разгледате форматирания документ на графичен дисплей преди да го отпечатате. Един пример е програмата *xdvi,* която показва в графичната среда на X, генериран от TEX"независим от устройството" файл. Други софтуерни приложения като *xfig,* предоставят WYSIWYG графичен интерфейс за рисуване на фигури и диаграми, които след това се описват с полиграфски език, така че да могат да бъдат включени във вашия текстов документ.

Съществуват много други програми, които са полезни при текстообработка. Например мощната система METAFONT се използва за създаване на шрифтове за TEX и е включена в адаптацията на TEX за Linux. Друга полезна програма е *ispell* - интерактивна програма за проверка на правопис. С ***makeindex*** могат да се създават индекси от LATEX документи. Предлагат се много пакети от макроси, базирани на *grojf* и TEX, с които могат да се форматират различни типове документи и математически текстове. Съществуват програми за превръщане на изходни текстове от формат T^X в *grqfj'n* обратно, а също и в много други формати.

Глава 9

В Глава 9 разглеждаме детайлно ШEX, *grojf* и други инструменти за форматиране на текст.

Платени приложения

Все повече разработчици на платен софтуер са заинтересовани от адаптирането на техните продукти за Linux. Адаптират се офис-пакети, приложения за създаване на документи (word processors), програми за администриране на мрежи и мощни системи за управление на големи бази данни. В първите две издания на тази книга не бе възможно да се напи-

ше много за предлагания платен софтуер. Но пазарът на Linux-софтуер се променя бързо и вярваме, че когато четете тази книга множество други платени софтуерни пакети ще бъдат адаптирани за Linux.

Star Division (от скоро част от Sun Microsystems) продават офис-пакета Star Office v5.x за Linux (съществуват версии за Windows 98, Windows NT и други Unix системи). Този пакет е повече или по-малко близък по функционалност до Microsoft Office и включва текстообработка, електронна таблица, редактор за HTML документи, модул за презентации и други приложения. Пакетът може да използва файлови формати от широка гама подобни приложения (включително и от Microsoft Office) и е безплатен, ако не се използва с търговска цел.

Corel адаптира пакета WordPerfect 8 за Linux. Той включва модули за текстообработка и работа с електронни таблици, софтуер за презентации и други. Пакетът е безплатен, ако не се използва с търговска цел. Цената на лиценз за един потребител е 49.95 щатски долара при закупуване от дистрибутор или 69.95 щатски долара при директна покупка от Corel.

Компании като Oracle, IBM, Informix, Sybase и Interbase предоставят (или са обявили планове за това) платени системи за управление на бази данни, работещи под Linux. IBM обяви бета-версия за Linux на своя сървър DB2 за управление на бази данни, която може да се изтегли безплатно от web-сайта на IBM *(http://www.ibm.com)*. В тази бета-версия няма реализация на clustering и някои други високи технологии, които съществуват в другите версии на DB2, но тя може да използва много процесори на съответните хардуерни платформи. IBM обяви, че ще предостави версия за Linux на известната мрежова файлова система AFS.

Caldera създаде пакета NetWare за Linux 1.0, който позволява Linux сървъри да поддържат услугите на NetWare за съхраняване на файлове, печат и NDS и да използват данни съвместно с Novell NetWare базирани сървъри. Продуктът е съвместим с NetWare 4.10b и NetWare-клиентите за Windows 3.1/95/98, DOS, Linux, Macintosh и UnixWare.

Езици и инструменти за програмиране

Linux предоставя пълна Unix среда за програмиране, включително всички стандартни библиотеки, инструменти, компилатори и програми за откриване и отстраняване на грешки, които можете да очаквате от една Unix система. При разработката на софтуер под Unix, приложенията и системни модули обикновено се пишат на езиците C и C++. Стандартният компилатор на Linux за C и C++ е *gcc* от проекта GNU. Това е модерен и мощен компилатор с много опции, който може да компилира и

програми, написани на C++ (поддържа се спецификация 3.0 на C++ от AT&T) или на Objective-C (друг обектно-ориентиран диалект на C).

Linux поддържа изцяло и сравнително новия програмен език Java. За Linux е адаптиран Java Development Kit v1.2 (известен още като "Java 2") на Sun. Java е обектно-ориентиран език и среда за работа, която поддържа разнообразни приложения, например аплети за web-страници, базирани на Интернет разпределени системи и програми, работещи директно със системи за управление на бази данни. Програмите, написани на Java, могат да работят на всяка машина (независимо от архитектурата или операционната система), която поддържа така наречената "виртуална машина" на Java. Много програмисти и крайни потребители са възхитени от възможностите, които предоставя Java.

Освен C, C++ и Java, за Linux са адаптирани много други компилирани и интерпретирани езици за програмиране, например Smalltalk, FORTRAN, Pascal, Lisp, Scheme и Ada, различни асемблери за писане на машинен код, любимия на Unix-хакерите Perl (скрипт-език, който е създаден, за да замени останалите скрипт-езици), Tcl/Tk (система, която използва команден език, подобен на езика на командните интерпретатори и предлага възможности за създаването на прости приложения под X Window) и Phyton (първият скрипт-език, който е обектно-ориентиран от самото си създаване).

За Linux е адаптирана и една от най-добрите програми за изчистване на грешки - *gdb*. *gdb* позволява трасирането на програма за откриване на грешки или причината за срив по време на работата й (за целта се използва соге файл - копие от паметта в момента на срива), *gprof* е програма за анализ на изпълнението на софтуерни приложения и дава статистика на изпълнението им, която ви позволява да откриете кои части от тях се изпълняват най-често и си струва да се оптимизират. Текстовият редактор Emacs предлага интерактивна среда за редактиране и компилиране, която можете да използвате за програмиране на различни езици. За Linux са адаптирани и инструментите GNU *make* и *imake,* които се използват за управление на компилирането на големи приложения. Освен това за Linux е адаптирана и системата RCS, с която можете да заключвате изходения код и да управлявате различните версиите на софтуера.

Linux е идеален за разработката на приложения за Unix. Той предоставя модерна среда за програмиране с всички удобства. Професионалните Unix програмисти и системни администратори могат да използват Linux за разработка на софтуер вкъщи и след това да го прехвърлят на Unix системите на работното си място. Това не само спестява много време и

пари, но и позволява работата да се извърши на удобно място.* Студентите по компютърни науки могат да използват Linux за изучаване на програмирането под Unix и за изследване на други аспекти на системата, например архитектурата на ядрото.

В Linux имате достъп не само до пълен комплект библиотеки и инструменти за програмиране, но и до целия изходен код на ядрото и библиотеките.

Глава 13
Глава 14

Глава 13 и Глава 14 са посветени на езиците и инструментите за програмиране под Linux.

Системата X Window

Системата X Window е стандартния графичен интерфейс за Unix машини. X е мощна среда, която поддържа много приложения. Когато използва X, потребителят може да има на екрана си едновременно много терминални прозорци, всеки от които съдържа различна сесия. В интерфейса на X обикновено се използват посочващи устройства като мишка.

Една от истински силните страни на системата X Window е изключителната й приспособимост към потребителските изисквания. Под X можете да емулирате интерфейса на други операционни системи - Windows 95/98, Macintosh, NeXTStep, както и да създадете изцяло нов интерфейс (много потребители на Linux предпочитат да създадат свой собствен интерфейс).

Написани са много приложения за X - игри, графични програми, инструменти за програмиране и документация и т.н. С Linux и X вашият компютър се превръща в истинска работна станция. Ако имате TCPЯP мрежа, можете дори да наблюдавате на своя екран X-приложения, работещи на други машини.

Системата X Window е създадена в MIT (Massachusetts Institute of Technology - Масачузетски институт за технологии) и се разпространява безплатно. Много фирми разпространяват разработени от тях допълнителни модули към оригиналния X-софтуер. Версията на X за Linux е известна като XFree86 (адаптация на X11R6) и се разпространява безплатно за базирани на PC Unix системи като Linux. XFree86 поддържа широка гама видео хардуер, включително VGA, Super VGA и повечето съвре-

Авторите използват Linux в домовете си са разработка и тест на приложения за X, които след това могат директно да се компилират на работни станции на други места.

менни видео-ускорители. XFree86 е цялостна реализация на X и съдържа самия X сървър, много приложения, полезни програми, библиотеки и документация.

Стандартните приложения за X включват *xterm* (терминален емулатор, в който се стартират повечето приложения с текстов интерфейс под X), *xdm* (програма за управление на потребителски сесии под X, с която можете да влезете в системата директно в графична среда), *xclock* (програма за показване на часовник), *xman* (базирана на X програма за разглеждане на справочни страници) и други. Приложенията за X под Linux са твърде много, за да бъдат изброени тук. Основната дистрибуция на XFree86 включва "стандартните" приложения от оригиналната дистрибуция на MIT. Много други приложения се разпространяват отделно и теоретично всяка програма за X би трябвало да се компилира без проблеми под Linux.

Външният вид на интерфейса на X се управлява в по-голямата си част от така наречения *window manager* (буквално "програма за управление на прозорци"). Тази програма се грижи за разполагането на прозорците на екрана, за начина, по който потребителя може да променя размера им, да ги намалява до "икони" или да ги премества, за вида на рамките на прозорците и т.н. Стандартната дистрибуция на XFree86 включва класическия window manager на MIT - *twm,* и някои усъвършенствани варианти, например Open Look Virtual Window Manager *{olvwm).* Особено популярен window manager сред потребителите на Linux е *fvwm2.* Той е малък и изисква по-малко от половината памет, необходима на *twm. fvwm2* има прозорци с тримерен изглед, както и виртуално работно пространство - ако потребителят придвижи мишката към края на екрана, цялото работно пространство се измества, симулирайки по-голям дисплей, *fvwm2* е изключително приспособим към потребителските изисквания. Повечето от функциите му са достъпни както от клавиатурата, така и чрез мишката. Много дистрибуции на Linux използват *fvwm2* като стандартен window manager.

Дистрибуцията на XFree86 съдържа библиотеки и заглавни файлове за програмистите, разработващи приложения за X. Поддържат се различни комплекти графични елементи (widgets), например Athena, Open Look и Xaw3D. Включени са всички стандартни шрифтове, икони, справочни страници и документация. Освен това се поддържат PEX (програмен интерфейс за тримерни графики) и Mesa (безплатна реализация на тримерните графични примитиви OpenGL).

В Глава 10, *Инсталиране на системата X Window,* и в Глава 11, *Настройка на графичната среда,* са разгледани инсталирането и използването на системата X Window в Linux системи.

KDE и GNOME

KDE и GNOME са два нови забележителни проекта, които предизвикват възхищение в света на Linux. И двете системи са предназначени за създаване на интегрирана графична работна среда, базирана на системата X Window. KDE и GNOME предлагат комплект мощни инструменти за разработка, библиотеки и приложения, които са интегрирани в една цялостна работна среда и обявяват следващата ера в работата с Linux на настолни системи. И двете среди имат богат графичен интерфейс, window manager, приложения и полезни програми, които са равностойни или превъзхождат възможностите на аналозите си в операционни системи като Windows 98.

С KDE и GNOME дори случайните и начинаещите потребители се чувстват удобно. Повечето дистрибуции по време на инсталирането си автоматично конфигурират едната от тези работни среди и потребителят може да работи изцяло в графична среда, без изобщо да му се налага да използва конзолата в текстов режим.

Макар че и KDE, и GNOME се стремят да направят Unix по-приятен за работа, те наблягат на различни неща. Основните цели на KDE са лекота при работа, стабилност и съвместимост на потребителския интерфейс с други графични среди (като Windows 95), докато в GNOME се набляга повече на добрия изглед и максималната гъвкавост.

Работа в мрежа

Linux е известен с една от най-мощните и най-бързи подсистеми за работа в мрежа в света. Тя е толкова добра, че все повече хора смятат, че Linux е чудесен избор за мрежов сървър. Linux поддържа двата основни мрежови протокола за Unix системи - TCP/IP и UUCP. TCP/IP (за любителите на съкращения - Transmission Control Protocol/Internet Protocol) е множество от конвенции, което позволява на системи в целия свят да комуникират помежду си, използвайки една единствена мрежа, известна като Интернет. С Linux, TCP/IP и връзка към Мрежата, можете да общувате с потребители и машини в Интернет чрез Web, електронна поща, групи по интереси в Usenet, да прехвърляте файлове с FTP и т.н.

Повечето TCP/IP мрежи използват Ethernet за физическо прехвърляне на данните. Linux поддържа много от разпространените Ethernet карти и интерфейси за персонални и преносими компютри. Linux поддържа също и Fast Ethernet, Gigabit Ethernet, ATM, ISDN, безжични мрежови контролери, Token Ring, packet radio и други високопроизводителни мрежови интерфейси.

Разбира се, не всеки има вкъщи Ethernet връзка с Интернет. Linux ви позволява да се свържете към Интернет чрез модем, като използвате протоколите PPP и SLIP. За да можете да използвате тези протоколи, вашият доставчик на Интернет трябва да ви предостави account на сървъра си, който да може да се използва през телефонна линия. Много фирми и университети предоставят входна точка за връзка чрез PPP или SLIP. А ако вашата Linux система има Ethernet връзка и модем, тя също може да се конфигурира като PPP/SLIP сървър за други машини.

Linux поддържа широка гама web-браузери (включително Netscape Navigator, безплатния му аналог Mozilla и браузера с текстов интерфейс Lynx). С web-браузер и Linux машина, свързана към Интернет, можете да сърфирате в Web, да изпращате email, да използвате chat, news и много други базирани на web услуги.

Освен това Linux поддържа много видове web-сървърй (включително известния безплатен Apache). Предполага се, че Apache под Linux управлява повече web-сайтове отколкото която и да е друга платформа в света. Apache е лесен за настройка и използване; ще ви покажем как можете да направите това в Глава 16, *Web и електронна поща.*

Глава 16

Samba е пакет, който позволява на Linux машини да работят като Windows сървър за файлове и печат. NFS позволява на системата ви да използва файлове съвместно с други компютри в мрежата. С NFS файловете на друг компютър изглеждат като част от файловете на вашата система. С FTP (File Transfer Protocol - протокол за прехвърляне на файлове) можете да прехвърляте файлове между отдалечени компютри.

За Linux са адаптирани всички основни Unix програми за e-mail и news - Elm, Pine, *rn, nn* и *tin.* Linux може да се конфигурира по всеки вкус за изпращане и получаване на e-mail и съобщения в Usenet.

Освен това Linux поддържа базираните на NNTP системи за електронни новини като C News и INN; програмите за прехвърляне на електронна поща *sendmail, exim* и *smail;* програмите *telnet, rlogin, ssh* и *rsh,* които позволяват на потребителя да влезе и да изпълнява команди в други машини в мрежата; и програмата *finger,* с която може да се получи инфор-

мация за други потребители в Интернет. Съществуват огромно количество базирани на TCPЯP протоколи и приложения.

Ако имате опит с TCPЯP приложения на други Unix системи, няма да се затрудните при работата си с Linux. Системата предоставя стандартен интерфейс за мрежово програмиране, базиран на гнезда (sockets), така че практически всяка програма, която използва TCPЯP, може да бъде адаптирана за Linux. X-сървърът на Linux също поддържа TCPЯP, което ви позволява да използвате графични приложения, работещи на отдалечени машини. Администрирането на мрежовата подсистема на Linux ще бъде познато на хората с опит в други Unix системи, защото програмите за настройка и наблюдение са подобни на аналозите си в BSD.

В Глава 15 описваме конфигурирането на TCPЯP и PPP под Linux. В Глава 16 разискваме конфигурирането на web-браузери, web-сървъри и софтуера за електронна поща.

UUCP (Unix to Unix Copy - копиране от Unix към Unix) е стар механизъм за прехвърляне на файлове. Обикновено UUCP машините са свързани една към друга през телефонни линии чрез модем, но същият протокол може да се използва и в TCPЯP мрежа. Ако нямате достъп до TCPЯP мрежа или PPP сървър, можете да конфигурирате системата си да изпраща и получава файлове и електронна поща чрез UUCP.

Телекомуникации и BBS софтуер

Телекомуникационният софтуер под Linux е много близък до този под Windows или друга операционна система. Всеки, който е използвал телекомуникационен пакет, ще открие, че еквивалентът му под Linux работи по подобен начин. Ако имате модем, ще можете да комуникирате с други машини като използвате един от телекомуникационните пакети за Linux. Много хора използват телекомуникационния софтуер за връзка с BBS и електронни комерсиални системи. Други използват модемите си за връзка с Unix машините на работното си място или в училище. Можете дори да използвате модема и Linux системата си за изпращане и получаване на факсове.

Един от най-популярните комуникационни пакети за Linux е Seyon. Той работи под X Window и предлага ергономичен интерфейс, който може да се настройва според нуждите на потребителя. Seyon има вградена поддръжка на различни протоколи за прехвърляне на файлове като Kermit, Zmodem и т.н. Други телекомуникационни програми са C-Kermit, *pcomm* и *minicom*. Те са подобни на комуникационните програми в другите операционни системи и са лесни за употреба. Zmodem за

Прилож.3

Linux е описан в Приложение 3, *Прехвърляне на файлове с протокола Zmodem*.

Ако нямате достъп до PPP сървър (виж предишния раздел), можете да използвате програмата *term* за мултиплексиране на серийна линия, *term* позволява да се отворят много потребителски сесии към отдалечена машина през модем. Освен това, *term* ви позволява да пренасочите връзката на отдалечен X клиент към локален X сървър през серийната линия, което ви дава възможност да използвате отдалечени приложения за X върху локалната Linux система. Друг софтуерен пакет, KA9Q, реализира подобен на SLIP интерфейс.

Ако нямате достъп до TCP/IP мрежа или UUCP връзка, Linux ви позволява да комуникирате с много BBS мрежи като Fidonet, с които можете да обменяте електронни новини и поща през телефонна линия.

Взаимодействие с Windows и MS-DOS

Съществуват различни програми за взаимодействие със света на Windows 95, Windows 98 и MS-DOS. Най-известното приложение е проектът Wine - емулатор на Microsoft Windows под X Window и Linux. Целта на този проекта, който все още се разработва, е да позволи изпълнението на приложения за Microsoft Windows под Linux. Подобна е и целта на Windows-емулаторът WABI на Sun Microsystems. Wine продължава да се разработва, като в момента може да изпълнява разнообразен софтуер за Windows, включително много настолни приложения и някои игри. Подробности за развитието на проекта можете да намерите на web-сайта *http://www.winehq.com*.

Linux предоставя удобен интерфейс за прехвърляне на файлове между Linux и Windows системи. Под Linux можете да монтирате Windows-дял или дискета и директно да работите с файловете на Windows по стандартния за Linux начин. Освен това, пакетите *mtools* и *htools* дават директен достъп до дискети, форматирани във формата съответно на MS-DOS и Macintosh.

Другоприложение е Linux емулатора на MS-DOS. Той ви позволява да изпълнявате много приложения за MS-DOS директно под Linux. Макар че Linux и MS-DOS са изцяло различни операционни системи, защитеният режим на процесорите x86 позволява определени задачи да се изпълняват в специален режим на емулация на 8086, благодарение на което можете директно да стартирате приложенията за MS-DOS.

Емулаторът на MS-DOS все още се разработва, но много от популярните приложения работят под него. Разбира са, приложения, които използват

странни и екзотични възможности на MS-DOS може никога да не получат поддръжка, защото това е само емулатор. Например, не трябва да очаквате поддръжка на програми, които използват възможностите на защитения режим на x86 като Microsoft Windows (в така наречения 386-enhanced mode).

Приложенията, които работят безпроблемно под емулатора на MS-DOS, включват 4DOS (команден интерпретатор), FoxPro 2.0, Harvard Graphics, MathCad, Turbo Assembler, Turbo C/C++, Turbo Pascal, Microsoft Windows 3.0 (в *реален* режим) и WordPerfect 5.1. Освен това с емулатора работят стандартните команди и програми за MS-DOS като *PKZIP*.

Емулаторът на MS-DOS е създаден специално като решение за тези потребители, които се нуждаят от MS-DOS само за някои приложения, но използват Linux за всичко останало. Емулаторът не е предназначен да бъде пълна реализация на MS-DOS. Разбира се, ако емулаторът не задоволява нуждите ви, винаги можете да стартирате MS-DOS на същия компютър, на който работи Linux (но не и едновременно с него). Програмата LILO (която се използва за зареждане на Linux) може да стартира избраната от потребителя операционна система при началното зареждане на компютъра. Linux може да съществува на една и същи компютър с други операционни системи, например с OS/2.

Ще ви покажем как можете да използвате емулатора на MS-DOS в раздела "DOS емулатори: Dosemu и xdos" в Глава 12, *Съвместимост с Windows и Samba,* заедно с другите начини за взаимодействие между Linux и Windows.

Други приложни програми

Съществуват множество разнообразни приложения за Linux, както може да се очаква от подобна операционна система. Основната цел на Linux в момента е персонална работа с Unix, но тази цел бързо се променя - нараства количеството на софтуера за бизнес и за научни цели, а разработчиците на платен софтуер създават все повече приложения за Linux.

Някои от приложенията с научна насоченост са следните: FELT (за анализ на крайни елементи), *gnuplot* (за създаване на графики и анализ на данни), Octave (пакет за символна математика, близък до MATLAB), *xspread* (калкулатор за електронни таблици), *xfractint* (адаптация за X на известния фрактален генератор Fractint), *xlispstat* (пакет за статистика) и т.н. Други интересни приложения са Spice (за създаване и анализ на електронни схеми) и Khoros (за обработка и визуализация на цифрови сигнали).

Разбира се, съществуват още много научни приложения, които са (или могат да бъдат) адаптирани за Linux. Не е никак трудно да адаптирате за Linux произволно Unix-приложение, независимо с коя област на науката се занимавате. Linux предоставя пълния интерфейс за програмиране под Unix, който може да се използва като база за създаване на научно приложение от всякакъв тип.

За Linux (като за всяка друга операционна система) съществуват разнообразни игри. Предлагат се класически игри с текстов интерфейс като Nethack и Moria (действието се развива в подземни лабиринти); MUDs (MultiUser Dungeons - игри за няколко човека с подобен сюжет) като DikuMUD и TinyMUD. Съществува огромно количество игри за X като *xtetris, netrek* и *Xboard* (X11 интерфейс за *gnuchess).* Нараства интересът към Linux и на производителите на платеии игри с интензивна графика - например ID Software създаде версии за Linux на Quake, Quake II, Quake III и Doom. Не са малко и другите създатели на игри за PC, които предлагат версии на игрите си за Linux, а много от тях дори разработват софтуера си почти изцяло под Linux!

За аудио-маниаците Linux поддържа голямо количество звукови карти и съответния софтуер като CDplayer (програма, която управлява CD-ROM като стандартен CD player), редактори за създаване на MIDI музика (които ви позволяват да композирате музика със синтезатор или друг, управляван с MIDI инструмент) и звукови ефекти.

Ако дистрибуцията ви не съдържа приложението, от което се нуждаете, можете да го потърсите в Интернет. Едно добро място, където се предлага безплатен софтуер, е Freshmeat *(http://www.freshmeat.net).* Списъкът в този сайт далеч не е завършен, но съдържа впечатляващ софтуер, по-голямата част от който работи под Linux. Хвърлете един поглед, дори само за да видите огромното количество програми, разработени за Linux.

Ако по никакъв начин не успеете да намерите адаптация за Linux на приложението, от което се нуждаете, винаги можете да опитате сами да го адаптирате от друга платформа за Linux. Или, в краен случай, можете да го напишете сами. Това е духът на Свободния Софтуер - ако искате нещо да бъде направено както трябва, направете го сами!

Авторски и сродните им права за Linux

Linux е защитен от документа, наречен GNU *General Public License,* или GPL. GPL, който понякога се нарича "copyleft", е разработен от Free Software Foundation за проекта GNU. GPL поставя няколко условия за разпространението на така наречения "free" софтуер. ДумаТа "free" се използва в смисъл "свободен", а не просто "безплатен". GPL винаги е бил разбиран погрешно и затова в този раздел ще направим опит да опишем достъпно смисъла и целта на GPL и значението му за Linux. Можете да намерите пълно копие на GPL на адрес:

http://www. gnu.org/copyleft/gpl. html

В началото Линус Торвалдс разпространява Linux с лиценз, който е по-ограничаващ от GPL и позволява софтуерът да се разпространява и променя свободно, но не позволява за него да се искат или получават пари. GPL позволява софтуерът да се продава и да се печели от него, но не разрешава по никакъв начин да се ограничават правата на купувачите от своя страна да разпространяват софтуера.

Първо трябва да подчертаем, че защитеният от GPL "свободен софтуер" *не е* публична собственост (public domain). Софтуерът, който е публична собственост, не е защитен от авторски права и, така да се каже, се притежава на обществото. От друга страна, софтуерът, защитен от GPL, принадлежи на авторите си. Това означава, че този софтуер се защитава от стандартните международни закони за защита на авторските и сродните им права и че авторът на софтуера е юридически определен. Това, че софтуерът може да се разпространява свободно, не означава, че е публична собственост.

Софтуерът, защитен от GPL, не е "shareware". Най-общо казано, "shareware" означава, че софтуерът е собственост на създателя си и потребителите трябва да заплатят пари за употребата на софтуера, макар че могат свободно да го разпространяват. За разлика от shareware, софтуерът, защитен от GPL, може да се използва безплатно.

GPL позволява на потребителите да променят софтуера и да разпространяват своите версии. Но всяка разработка, за основа на която се използва софтуер, защитен от GPL, задължително трябва да се разпространява със същия лиценз. С други думи, не е разрешено някоя компания да промени Linux и след това да го разпространява с по-ограничаващ лиценз. Всеки софтуер, който произлиза от Linux, трябва да се разпространява с GPL.

Частни лица и организации могат да разпространяват GPL софтуер срещу заплащане и дори да печелят от това. Но дистрибуторът не може да отнеме тези права от купувача, тоест ако някой закупи от някакъв източник GPL софтуер, след това също може да го разпространява безплатно или срещу заплащане.

На пръв поглед се получава противоречие. Защо GPL софтуера се продава, след като може да бъде получен безплатно? Отговорът е прост. При закупуването на свободен софтуер, заплащате не самия софтуер, а носителят (например компакт-диск) и трудът за създаването на конкретната дистрибуция и разпространението й. Това е разрешено от GPL.

Организациите, които продават свободен софтуер, трябва да спазват определени изисквания на GPL. Първо, те не могат да ограничават правата на потребителите, които са закупили софтуера. Това означава, че ако закупите компакт-диск с GPL софтуер, можете да го копирате и разпространявате безплатно или дори да го препродадете. Второ, разпространителите трябва ясно да покажат, че софтуерът е защитен от GPL. Трето, дистрибуторите са длъжни да предоставят безплатно пълния изходен код на разпространявания от тях софтуер или да насочат потребителите към място, откъдето той може да бъде получен безплатно. Това позволява на всеки, който закупи GPL софтуер, да прави промени в него.

Условието от GPL, разрешаващо на компаниите да разпространяват и продават софтуера, е много полезно. Не всеки има достатъчно добър (и евтин) достъп до Интернет, за да може да изтегли софтуер като Linux. GPL позволява на компаниите да разпространяват и продават софтуер на тези, които нямат безплатен (или на приемлива цена) достъп до софтуера. Например, много организации продават Linux на дискети или компакт-дискове с пощенска заявка и печелят от тази услуга. Създателите на Linux могат да не получават нищо от тази печалба (това се подразбира, щом софтуерът се разпространява с GPL). С други думи, Линус Торвалдс знае, че компаниите могат да печелят от Linux, но самият той да не получи и една стотинка от това.

В света на свободния софтуер важното не са парите. Целта на свободния софтуер винаги е била да се разработват и разпространяват фантастични продукти и всеки да може да ги получи и използва. В следващия раздел описваме как това влияе върху разработката на Linux.

Отвореният Код и философията на Linux

Често новите потребители на Linux не разбират някои неща и имат неправилни очаквания за системата. Linux е уникална операционна система и за ефективното й използване е важно да се разберат нейните философия и дизайн. В центъра на философията на Linux е концепцията за Софтуер с Отворен Код.

Отворен Код е термин, отнасящ за софтуер, чийто изходен код (реализацията на алгоритмите, по които работи програмата) е безплатен, достъпен за всеки и може свободно да се променя и разпространява. Софтуерът, защитен от GNU GPL, описан в предишния раздел, попада в категорията отворен код. Не е изненадващо, че в тази категория попада и голямо количество софтуер, защитен от други, но подобни на GPL лицензи. Например всеки софтуер, който може свободно да се променя, но няма същите стриктни изисквания за разпространение като GPL, също се смята за Отворен Код.

Така наречените "модел за разработка Отворен Код" е феномен, започнат от Free Software Foundation и популяризиран от Linux. Това е изцяло различен начин за създаване на софтуер, който разкрива всяка страна на разработката, изчистването на грешки, тестването и изучаването на софтуера за всеки, който се интересува от това. Вместо да се разчита на единствена корпорация за разработката и поддръжката на част от софтуера, Отвореният Код позволява софтуерът да се развива отворено в общество от разработчици и потребители, които са мотивирани от желанието *да създават добър софтуер,* вместо просто да печелят пари.

Книгата *Open Sources* на издателството O'Reilly & Associates, Inc. може да се използва като въведение в модела за разработка Отворен Код. Тя съдържа статии от водещи разработчици (включително Линус Торвалдс и Ричард Сталман - основателят на Free Software Foundation) за процеса на създаване на приложения с отворен код и е редактирана от Chris DiBona, Sam Ockman и Mark Stone.

На отворения код се обръща голямо внимание от медиите. Някои автори го наричат "следващата вълна" в разработката на софтуер, която ще помете стария начин на създаване на програми, при който не се разпространява изходния им код. Бъдещето ще покаже дали това ще се случи, но вече се случиха някои обнадеждаващи събития, които показват, че това не е невъзможно. Например, Netscape Corporation публикува кода на своя web-браузър като проект с отворен код, наречен Mozilla, а компа-

нии като Sun Microsystems, IBM и Apple обявиха, че ще разпространят някои свои продукти като Отворен Код с надеждата, че те ще се развият с усилията на общества от софтуерни разработчици.

На отворения код се обръща голямо внимание от медиите като Linux е в центъра на вниманието им. За да се разбере начинът на мислене при разработването на Linux, е полезно да се разгледа създаването на една традиционна Unix система.

Във фирмите-разработчици на платени варианти на Unix, цялата система се създава при строги правила за контрол на качеството като се използват специални системи за управление на разработката и промените в софтуера и документацията и за откриване и поправяне на грешки. На разработчиците не е позволено да добавят нови възможности на системата или да променят ключови части от кода в нея по собствено желание - промените трябва да се правят само за отстраняване на открита грешка и впоследствие да се отбележат в системата за управление на разработката, така че при необходимост да могат да бъдат премахнати. На всеки разработчик се определя една или повече части от кода на системата и само той може да ги променя, докато той се грижи за този код.

Отделът за контрол на качеството извършва серии от сериозни тестове (така наречените "обратни тестове") на всяка нова версия на операционната система и докладва откритите грешки. Разработчиците трябва да поправят тези грешки след откриването им. Използват се сложни системи за статистически анализ, за да се гарантира, че определена част от грешките е поправена преди разпространяването на следващата версия и че операционната система отговаря на определени критерии за качество.

Процесът, използван от създателите на комерсиални Unix системи, е много сложен и добре обмислен. Компанията трябва да има количествени доказателства, че следващата версия на операционната система е готова за разпространение и затова се събират и обработват на статистически данни за качеството на работата й. Разработката на комерсиални Unix системи е трудоемък процес, в който често участват стотици (ако не и хиляди) програмисти, изпитатели, документатори и административен персонал. Разбира се, няма два еднакви комерсиални разработчици на Unix, но от казаното дотук трябва да сте добили обща представа за начина, по който се работи в тези компании.

Linux се разработва по съвсем друг начин, нямащ нищо общо с принципите за организирана разработка, системите за управлението й и доклад-

ването на грешките или статистически анализ. Linux е, и вероятно ще продължи да бъде, хакерска операционна система.

Linux е разработен основно с общите усилия на доброволци от цял свят. Не съществува единна организация, която да се грижи за разработката на системата. По-голямата част членовете на Linux обществото използват за комуникация помежду си Usenet и различни web-сайтове. Все пак, съществуват няколко конвенции за начина на разработка. Например, ако някой програмист желае написан от него код да бъде включен в "официалното" ядро, трябва да го изпрати на Линус Торвалдс. Линус ще го изпробва и, ако кодът не създава проблеми и спазва общите правила за архитектурата на системата, ще го включи в ядрото.

Самата система има гъвкава архитектура с богати възможности. Макар че напоследък намалява броят на съществените промени, общото правило е на всеки няколко седмици да се разпространява нова версия на ядрото (понякога дори по-често). Разбира се, това е твърде условно и зависи от различни фактори, като например броят на грешките, които трябва да бъдат поправени, получената информация от потребителите, които тестват предварителните версии на кода и количеството сън, което е необходимо на Линус през тази седмица.

Разбира се, между версиите остават грешки и проблеми. След като се отстранят критичните и често срещани грешки, системата се смята за "стабилна" и се разпространява нова версия. Основната цел в разработката на Linux не е да се създаде перфектен безгрешен код, а да се създаде безплатна реализация на Unix. Linux е преди всичко система за разработчиците.

Всеки, който е създал нова възможност на ядрото или ново приложение за системата, обикновено го разпространява в така наречената "алфа"-версия. Това е етап от тестването на продукта, в който обикновено се включват само най-смелите потребители, които искат да изчистят проблемите в първоначалния код. Понеже Linux обществото е базирано основно на Интернет, алфа-версиите на софтуера обикновено се разполагат на един или повече web-сайта, свързани с Linux (виж Приложение А), след което се изпраща съобщение до една или повече групи в Usenet с описание откъде да се изтегли и как да се изпробва кода. Потребите-

+ "Хакер" означава човек, отдал се на програмирането и получаващ удоволствие от използването на компютрите и правенето на интересни неща с тях. Това значение е различно от разпространената концепция за вредител и нарушител на закона.

лите, които изтеглят и тестват алфа-версията на софтуера, могат след това да изпратят резултати, корекции на грешки и въпроси към автора.

След като се отстранят първоначалните проблеми в кода, той се смята за "бета-версия". Това означава, че кодът е сравнително стабилен, но не е завършен (т.е. работи, но може би някои възможности не са реализирани). Възможно е също софтуерът да премине направо към финален етап, в който случай се смята за завършен и използваем. Ако кода е за ядрото, след завършването му, разработчика го изпраща на Линус с молба да га включи в стандартното ядро или като опционален модул към него.

Имайте в предвид, че това са само конвенции, а не правила. Някои програмисти са достатъчно уверени в софтуера си и не разпространяват алфа или тестова версия. Подобно решение винаги е въпрос на личен избор на разработчика.

Какво се случва със сериите от тестове и строгия контрол на качеството? Те се заменят от ранното и често разпространяване на версии. Потребителите са най-добрите изпитатели на софтуера, защото го изпробват на разнообразни системи едновременно с множество реални софтуерни приложения, а това трудно може да се постигне от която и да е група за контрол на качеството. Една от най-добрите страни на този начин за разработка и разпространение е, че грешките (и пропуските в сигурността) често се откриват, докладват и поправят за *часове,* а не за дни или седмици.

Може би ви учудва факта, че изобщо е възможно да бъде направено нещо от една толкова хаотична група от разработчици-доброволци, които при това създават и изчистват от грешки една Unix система. Оказва се, че това е един от най-ефективните и мотивирани начини за разработка. Цялото ядро на Linux е написано *от нулата,* без да се използва какъвто и да е код от фирмени източници. След това доброволци са положили големи усилия за адаптирането на целия свободен софтуер под слънцето, така че да може да работи под Linux. Освен това са написани и адаптирани множество библиотеки, разработени са файлови системи и хардуерни драйвери за много от популярните устройства.

Софтуерът за Linux обикновено се разпространява като *дистрибуция.* Тя се състои от пакети предварително подготвен софтуер, които образуват цялостна система. За повечето потребители би било много трудно да изградят завършена система, започвайки от самото начало - първо да компилират ядрото, след това да добавят полезни програми и да инсталират останалия софтуер ръчно. Вместо това потребителите могат да използват някоя от готовите софтуерни дистрибуции, включваща всичко необходимо за инсталиране и използване на системата. Не съществува стан-

дартна дистрибуция - съществуват много дистрибуции, всяка от които има силни и слаби страни. В тази книга описваме как се инсталират дистрибуциите на Red Hat, SuSE и Debian, но материалът може да ви помогне при инсталирането на произволна дистрибуция.

Независимо от пълнотата на софтуера за Linux, все пак трябва да имате малко знания за Unix, за да можете да инсталирате и използвате една цялостна система. Не съществува напълно изчистена от грешки дистрибуция на Linux и затова понякога се налага да поправите ръчно някои дребни грешки след инсталацията.

Съвети за начинаещи потребители на Unix

Инсталирането и използването на собствена Linux система не изисква голям опит в работата с Unix системи. Всъщност, много от начинаещите потребители на Unix успешно инсталират Linux на компютрите си. Това е полезен опит за изучаването на системата, но може да ви се стори много тежко, ако възникнат проблеми. Ако имате късмет, ще можете да инсталирате и да започнете да използвате Linux, дори ако нямате никакви познания за Unix. Все пак, съществуват и по-сложни проблеми при използването на Linux - инсталиране на нов софтуер, компилиране на ядрото и т.н. За тяхното решаване са необходими някои основни познания за Unix. (Все пак трябва да отбележим, че повечето дистрибуции на Linux се инсталират и конфигурират не по-трудно от Windows 98 и определено по-лесно от Windows NT).

За щастие, като използва собствена Linux система, ще можете да научите основите на Unix, които са необходими за решаването на тези задачи. Тази книга съдържа достатъчно информация, която ще ви помогне да започнете. Глава 4 е ръководство, което описва основите на Unix, а Глава 5 съдържа информация за системното администриране на Linux. Може би най-добре е да прочетете тези глави преди да се опитате да инсталирате Linux - информацията в тях е безценна при разрешаването на евентуалните проблеми при инсталирането на системата.

Никой не може да очаква за една нощ да се превърне от начинаещ потребител на Unix в опитен системен администратор. Освен това, не мислете, че съществува реализация на Unix, която в никакъв случай няма да създаде проблеми. Трябва да сте подготвени за приключението, което ви очаква. В противен случай, ако сте начинаещ потребител на Unix, е възможно да бъдете много разочарован от системата.

Съвети за опитни потребители на Unix

Дори хората, които имат многогодишен опит в програмирането и системното администриране на Unix-системи, понякога се нуждаят от малко помощ, преди да инсталират и започнат да използват Linux. Съществуват особености на системата, които трябва да научат дори опитните потребители на Unix, преди да започнат свободно да използват Linux. От една страна, Linux не е платена Unix система. Тя не се опитва да спазва стандартите, с които може би сте свикнали при работата си с друга реализация на Unix. В известен смисъл, Linux *предефинира* света на Unix, като дава възможност на останалите системи да заслужат парите си. За да бъдем по-конкретни, макар че стабилността е важен фактор за Linux, тя не е *единственият* фактор.

Може би по-важна е функционалността на системата. В много случаи към стандартното ядро на системата се добавят нови модули, макар че те не са изцяло изчистени от грешки и функционално завършени. Предполага се, че по-важно е кодът да бъде достъпен на потребителите, така че те да могат да го използват и тестват, вместо да се забави разпространението му до "пълното" му завършване. Почти всички софтуерни проекти с Отворен Код имат алфа-версия, която се разпространява, преди да бъде изцяло тествана. По този начин всеки може да тества и да усъвършенства кода, а този който го смята за достатъчно добър, може директно да го използва. Разработчиците на платени реализации на Unix изключително рядко разпространяват софтуер по този начин.

Ако сте били повече от десет години системен администратор на Unix системи и сте работили с всяка платена реализация на Unix под слънцето (неволна игра на думи; Sun - "слънце" на английски - е името на един от най-големите разработчици на Unix системи в света), отделете известно време и на Linux. Системата е много модерна и динамична. На всеки няколко месеца се разпространява нова версия на ядрото, освен това непрекъснато се създават нови програми за Linux. Днес системата ви може да е напълно в течение със съвременните тенденции, а утре да изглежда като находка от Каменната Ера.

След като всичко се развива толкова динамично, как можете да сте в течение с непрекъснато променящия се свят на Linux? В повечето случаи най-доброто решение е постепенно да осъвременявате системата. Това означава да актуализирате само тези части от системата, които се *нуждаят* от осъвременяване и то тогава, когато смятате това за необходимо. Например, ако никога не използвате Emacs, няма голям смисъл да инсталирате всяка нова версия на Emacs на компютъра си. Дори тези, които непрекъснато използват Emacs, нямат причина непрекъснато да го акту-

ализират, освен ако не открият в новата версия полезна за работата си възможност. Трудно е да се посочат причини, поради които системата трябва да има винаги най-новия софтуер.

Надяваме се, че Linux отговаря или надминава очакванията ви за домашна Unix-система. В сърцевината на Linux е духът на свободния софтуер, на непрекъснатата разработка и развитие. Linux обществото предпочита разширяването пред стабилността. Много хора трудно приемат такъв подход, особено тези, които са свикнали със света на комерсиалния Unix. Не трябва да очаквате Linux да бъде съвършен - нищо не е съвършено в света на свободния софтуер (или в който и да е друг свят). Все пак, ние смятаме, че Linux наистина е завършен и използваем поне колкото другите реализации на Unix.

Разлики между Linux и другите операционни системи

Важно е да се разбира разликата между Linux и операционни системи като Windows 95/98, Windows NT, OS/2 и други реализации на Unix за персонални компютри. На първо място, трябва да е ясно, че Linux може да работи прекрасно с други операционни системи на един и същи компютър. Това означава, че можете безпроблемно да използвате Windows NT или OS/2 на компютър, на който е инсталиран Linux. Съществуват дори начини за взаимодействие между различните операционни системи, които ще разгледаме по-късно.

Защо Linux?

Защо да се използва Linux вместо платена операционна система? Възможно е да се посочат хиляди причини. Една от най-важните е, че Linux е отличен избор за персонална работа с Unix. Ако сте разработчик на софтуер за Unix, защо трябва вкъщи да използвате Windows? Linux ви позволява да разработвате и тествате софтуер за Unix на домашния си компютър, включително програми за работа с бази данни и приложения за X. Почти всеки университет използва Unix в своята компютърна система. С Linux можете да работите на собствена Unix система, която да настроите според собствените си нужди. Инсталирането и използването на Linux е чудесен начин за изучаване на Unix за тези, които нямат достъп до други Unix машини.

Но нека не се отклоняваме. Linux е подходящ не само за професионални потребители на Unix. Той е достатъчно мощен и развит, за да може да

изпълнява сложни задачи и да се използва като база за разпределени системи. Много фирми преминават на Linux от други базирани на Unix системи. Linux има отлично съотношение цена/производителност. Той е една от най-стабилните и мощни операционни системи в света. Благодарение на Отворения Код, потребителят може изцяло да приспособи Linux към нуждите си. Университетите използват Linux като основа за курсове по архитектура на операционните системи. Големите производители на софтуер започват да оценяват възможностите, които предоставя тази безплатна операционна система.

Linux срещу Windows 95 и Windows 98

Нерядко на един и същи компютър се използват едновременно Linux и Windows 95/98. Много потребители на Linux разчитат на офис приложения, които работят под Windows. Макар че Linux предоставя аналогични програми (например LJX) и количеството платен софтуер за Linux непрекъснато нараства, съществуват множество причини, поради които определени потребители искат да използват Windows съвместно с Linux. Например не винаги е лесно да се прехвърлят в TLX или друг формат големи документи (например дисертация), които са написани с Microsoft Word (макар че пакетът Star Office за Linux вероятно ще се справи). Съществуват много платени приложения за Windows, които нямат версия за Linux и няма причина, поради която да не използвате и двете операционни системи.

Както може би знаете, Windows 95 и Windows 98 не използват пълноценно възможностите на процесорите от фамилията Intel x86. От друга страна, Linux работи изцяло в защитен режим и използва всички възможности на машината, включително и на многопроцесорните платформи.

Можем дълго да дискутираме плюсовете и минусите на Windows и Linux. Все пак, достатъчно е да спомена, че двете системи са от съвсем различен тип. Windows е сравнително евтин (в сравнение с други платени операционни системи) и е много разпространен в света на персоналните компютри. Няма друга операционна система, която да е достигнала нивото на популярност на Windows, основно защото останалите операционни системи са с недостъпна цена за повечето потребители. Много малко потребители на персонални компютри могат да си позволят да похарчат хиляди долари само за операционната система. От друга страна, Linux е безплатен и вече всеки има възможност да избира.

Разбира се, вие сами трябва да направите преценка на Linux и Windows в зависимост от очакванията и нуждите си. Linux не е за всеки. Но ако винаги сте искали да имате пълна Unix система вкъщи, при това без да заплащате високата цена на другите реализации на Unix за персонални компютри, Linux може би е това, което търсите.

Съществуват програми, които позволяват взаимодействие между Linux и Windows. Например, от Linux лесно могат да се четат Windows файлове. Освен това продължава разработването на Wine - емулатор на Windows, който би позволил на Linux да изпълнява много от популярните приложения за Windows.

Linux срещу Windows NT

В света на персоналните компютри се създават и други мощни операционни системи. В областта на сървърите все по-популярен става Microsoft Windows NT.

Windows NT, също като Linux, е истинска многозадачна операционна система, която поддържа многопроцесорни платформи, различни процесорни архитектури, виртуална памет, работа в мрежа, има вградена система за сигурност и т.н. Съществената разлика между Linux и Windows NT е, че Linux е версия на Unix и затова може да използва голяма част от разработките на Unix-обществото.

Съществуват много реализации на Unix от различни производители. В Unix-обществото има голям натиск за стандартизация във формата на отворени системи, но нито една компания не контролира този дизайн. Затова всеки производител (или както се оказва - всеки хакер) може да реализира тези стандарти за дадена реализация на Unix.

От друга страна, Windows NT е собственост на корпорацията Microsoft и тя контролира интерфейса и архитектурата на системата. Само Microsoft може да прави промени в Windows NT (така че не очаквайте в близко бъдеще да видите безплатна версия на Windows NT). От една страна, това е добре за потребителите - съществуват точни стандарти за програмирането и потребителския интерфейс за разлика от стандартите в отворените системи. Навсякъде по света NT е NT.

Изглежда много вероятно в близките години Linux и Windows NT да се борят за свой дял в пазара на сървъри. Зад Windows NT стои цялата мощ на маркетинговата машина на Microsoft, докато Linux притежава общество от хиляди разработчици, които помагат на системата да се развива чрез модела Отворен Код. Засега тестовете показват, че и двете системи имат и силни, и слаби страни. Linux печели убедително в доста области,

като най-забележима е разликата в работата в мрежа. Освен това, Linux е много по-малък от Windows NT, има много по-добро съотношение цена/производителност и изглежда по-стабилен. (Докато за Windows NT се смята, че се срива доста често, Linux-машини работят без прекъсване в продължение на месеци). Може би изглежда учудващо, че "малкият" Linux е толкова сериозен конкурент на Microsoft, но това не е изненада за тези, които знаят колко ефективен всъщност е процесът на разработка с Отворен Код.

Други реализации на Unix

Съществуват няколко други реализации на Unix за персонални компютри. Архитектурата на фамилията Intel x86 е подходяща за Unix и някои софтуерни производители се възползват от това - например Sun (със Solaris x86), SCO и BSD.

Другите реализации на Unix за PC имат доста близки възможности с тези на Linux. Ще откриете, че почти всички платени версии на Unix поддържат практически същия софтуер, среда за програмиране и мрежови възможности. Все пак съществуват някои важни разлики между Linux и платените версии на Unix. Те се дължат преди всичко на произхода на Linux - той е създаден като "персонална" Unix система, а не като операционна система, която ще работи на големи сървъри (макар че Linux работи безупречно и в двата случая).

На първо място, Linux поддържа много по-широк спектър хардуер отколкото поддържат другите реализации на Unix, просто защото за Linux е важно да работи с всевъзможни модели звукови, графични, мрежови и SCSI контролери. Освен това, всеки, който има достатъчно време и желание, може да напише драйвер за някой конкретен контролер и да го публикува като Отворен Код. В следващия раздел ще разгледаме хардуерните изисквания на Linux.

За много потребители един от най-важните фактори при избора на софтуер е цената. Софтуерът за Linux е безплатен за всеки, който има достъп до Интернет (или друга компютърна мрежа) и може да го изтегли. Освен това, софтуерът може да се закупи и получи по пощата на компакт-диск, като често доставката включва документация и се осигурява телефонна поддръжка. Разбира се, всеки може да копира Linux от приятел или да плати част от цената, ако закупи софтуера с някой друг. Ако искате да инсталирате Linux на много компютри, достатъчно е да закупите само едно копие на софтуера. Linux не се разпространява с "лиценз само за една машина".

Не бива да се подценява значението на платените реализации на Unix - заедно с цената на самия софтуер, потребителят плаща за документация и поддръжка и получава гаранция за качество. Това са важни фактори за големите институции, но потребителите на персонални компютри обикновено не се нуждаят от тях. Вече няколко компании, например Red Hat и LinuxCare, предлагат платена поддръжка за Linux. Caldera (друг дистрибутор на Linux) предлага денонощна поддръжка. Много фирми и университети смятат, че за предпочитане е да използват Linux с персонални компютри вместо Unix със скъпи работни станции. Linux предоставя функционалността на работна станция като използва евтин хардуер.

Съществуват и други безплатни или евтини реализации на Unix за x86. Една от най-разпространените е FreeBSD - реализация и адаптация на BSD Unix за фамилията x86. FreeBSD е сравнима с Linux в много отношения и отговорът на въпроса коя от двете е "по-добра" зависи от личните нужди и очаквания на потребителя. Единствената важна разлика е, че Linux се разработва отворено (и всеки желаещ може да помогне в процеса на разработката), докато FreeBSD се разработва в затворен екип от програмисти. Това е причината да съществуват сериозни философски и архитектурни различия между двата проекта. Целите им са изцяло различни - целта на Linux е създаването на цялостна Unix-система, като се започне на чисто, а целта на FreeBSD е да се промени съществуващият код на BSD, така че да работи на x86.

NetBSD е друга адаптация на дистрибуцията BSD NET/2 за множество платформи, включително и за x86. NetBSD е с малко по-отворена структура от FreeBSD и е сравнима с нея в много отношения. OpenBSD е още една версия на BSD.

Друг проект, който си струва да отбележим, е HURD. Той се разработва от Free Software Foundation и целта му е да се създаде и разпространи свободна версия на Unix за много платформи. Повече информация за HURD можете да получите от Free Software Foundation. Този проект все е още в начален етап на разработка и интересът към него намалява и се измества към Linux.

Съществуват и други евтини версии на Unix - например Minix (академичен, но полезен клон на Unix, на който е базирана началната разработка на Linux). Някои от тези реализации са интересни преди всичко от академична гледна точка, докато други са пълноценни системи, подходящи за реална дейност. Но много от потребителите на персонален Unix се прехвърлят на Linux.

Хардуерни изисквания

Предполагаме, че вече сте се убедили колко добра система е Linux и какви прекрасни неща може да направи за вас. Все пак, преди да се втурнете да инсталирате софтуера, трябва да знаете какви са хардуерните изисквания и ограничения на Linux.

Не забравяйте, че Linux е разработен от потребителите си. Това означава, че повечето хардуер, който се поддържа от Linux, е този, до който имат достъп потребителите и разработчиците на системата. Linux поддържа повечето популярен хардуер и периферни устройства за системите 80x86 (всъщност, Linux вероятно поддържа повече хардуер от която и да е платена реализация на Unix). Все пак, някои неизвестни и екзотични устройства, както и тези, за които не се публикуват спецификации от създателите им, все още не се поддържат. Непрекъснато расте количеството на поддържания хардуер, така че ако любимите ви устройства все още не се поддържат, вероятно това ще стане скоро.

Поддържането на хардуер под Linux е затруднено от това, че много компании са решили да запазят хардуерния интерфейс на устройствата си като частна собственост. Като резултат от това, разработчиците на Linux просто не могат да напишат драйвери за тези устройства (а ако успеят, драйверите ще бъдат собственост на компанията, притежаваща интерфейса, което нарушава GPL). Компаниите, които не публикуват интерфейса на устройствата си, пишат свои собствени драйвери за операционните системи (например за Microsoft Windows), затова на крайния потребител (това сте вие) не е необходимо да познава интерфейса. За съжаление, това не позволява на разработчиците на Linux да напишат драйвери за тези устройства.

Много малко може да се направи, за да се подобри тази ситуация. В някои случаи програмистите се опитват да пишат хакерски драйвери, които са базирани на предположения за интерфейса. В други случаи, разработчиците работят с компанията по въпроса, като се опитват (с променлив успех) да получат информация за интерфейса на устройството.

Linux има някои специфични възможности за работа с преносими компютри, например поддръжка на PCMCIA (или "PC Card") и APM. Пакетът с инструменти за PCMCIA за Linux включва драйвери за много PCMCIA устройства, включително модеми, Ethernet карти и SCSI контролери. Документът PCMCIA HOWTO съдържа информацията, която ще ви е необходима в началото.

APM позволява на ядрото да следи напрежението на батериите в преносимия компютър и да извърши необходимите действия (като автоматич-

но започване на процедурата по изключване), когато напрежението спадне. APM позволява на централния процесор да работи в специален режим с малка консумация на ток, когато компютъра не се използва. Всичко това е лесно за конфигуриране като опция на ядрото. Съществуват различни програми за взаимодействие с APM, например *apm* (показва информация за състоянието на батериите) и *apmd* (съставя статистика на състоянието на батериите и може да се използва за иницииране на специални процедури при промяна на състоянието им). Тези програми са включени в повечето дистрибуции на Linux.

[68] Hardware HOWTO

В следващите раздели ще се опитаме да обобщим хардуерните изисквания на Linux. Документът Linux Hardware HOWTO (виж раздела "Източници на информация за Linux" по-долу в тази глава за описание на документите HOWTO) съдържа по-пълен списък на хардуера, който се поддържа от Linux.

ЗАБЕЛЕЖКА Много от драйверите за Linux са в етап на разработка. Някои дистрибуции могат да включват драйвери с експериментални възможности. В този раздел са изброени предимно устройства, които се поддържат от известно време и драйверите им се смятат за стабилни. Когато не сте сигурни за някой конкретен драйвер, можете да се консултирате с документацията за дистрибуцията на Linux, която използвате (за повече информация за дистрибуциите на Linux виж раздела "Дистрибуции на Linux" в Глава 2, *Подготовка за инсталиране на Linux).*

Глава 2

Още едно предупреждение - понякога доставчиците на хардуер заменят някои системни компоненти (например мрежовата платка) с по-нов модел, независимо от първоначалната ви заявка. Ако се съмнявате, проверете точно какъв хардуер имате.

Изисквания за дънна платка и процесор

В момента Linux поддържа системи с процесори Intel 80386, 80486, Pentium, Pentium Pro, Pentium II и Pentium III. Това включва всички варианти на този тип процесори, например 386SX, 486SX, 486DX и 486DX2. Поддържат се и системи с "клонинги" на тези процесори, произведени от AMD и Cyrix.

Linux е адаптиран за някои архитектури, които не използват процесори на Intel, включително Alpha AXP, MIPS, PowerPC, SPARC и Motorola 68k. В момента някои от тези адаптации са зрели от други. Red Hat и De-

bian разпространяват версии на дистрибуциите си за SPARC и Alpha като допълнение на версията за Intel x86. SuSE има версия за Alpha, а Debian дори предлага версия на дистрибуцията си за Motorola 68k (виж Приложение Д, *Инсталиране на Linux/m68k върху системи с процесор от серията Motorola 68000).* В тази книга обръщаме най-голямо внимание на версията на Linux за системи Intel x86. Основната част на книгата (с изключение на разделите, в които се описват хардуерните изисквания и началното инсталиране) е приложима за адаптациите на Linux за всички архитектури.

Прилож.Д

Ако имате стар процесор като 80386 или 80486SX, бихте могли да използвате математически копроцесор, макар че това не е задължително (ако нямате математически копроцесор, ядрото на Linux го емулира). Поддържат се всички стандартни копроцесори като IIT, Cyrix FasMath и копроцесорите на Intel.

Дънната платка трябва да използва някоя от следните архитектури на шината: ISA, EISA, PCI или MicroChannel (MCA). Архитектурата дефинира начина, по който процесорът взаимодейства с периферните устройства и други компоненти, свързани към шината.

Linux поддържа и системите, които използват локална шина (за по-бърз достъп до видео-паметта или твърдия диск). Препоръчва се да използвате стандартна архитектура на локалната шина, например VESA Local Bus (VLB).

Изисквания за паметта

Linux може да работи със съвсем малко памет в сравнение с другите модерни операционни системи. Бихте могли да използвате Linux дори с 8 MB RAM памет, но ви препоръчваме да имате поне 16 MB. Колкото повече памет имате, толкова по-бързо ще работи системата.

Linux поддържа пълното 32-битово адресиране на процесорите 80x86. С други думи, той ще използва автоматично цялата RAM памет. За по-старите версии на ядрото е необходимо да се укаже с параметър колко е паметта в компютъра, ако имате повече от 64 MB.

Linux ще работи безпроблемно дори с 8 MB памет, включително X Window, Emacs и т.н. Все пак, да имате повече памет е важно почти толкова, колкото да имате по-бърз процесор. Ако само вие използвате компютъра, 16 мегабайта са достатъчни. Ако предвиждате със системата да работят много потребители, трябва да имате поне 32 MB. Linux може да поддържа платформи с много големи количества памет - 1 GB или повече.

Повечето потребители на Linux отделят част от твърдия си диск като място за swap (буквално "място за замяна"), което се използва за реализиране на виртуална памет. Бихте могли да използвате виртуална памет, дори ако имате в компютъра си голямо количество физическа памет. Макар че използването на swap не може да замени истинската физическа памет, то позволява на системата ви да изпълнява по-големи приложения, като на диска се прехвърлят неизползваните в момента части от кода. Размерът на мястото за swap зависи от множество фактори. Този въпрос се разглежда в раздела "Изисквания за дяловете на Linux" в Глава 2.

Изисквания за контролера на твърдия диск

За да използвате Linux, не е необходимо да имате твърд диск. Всъщност, за да използвате Linux система с минимални възможности, ви е достатъчна само една дискета! Разбира се, стандартният начин за работа с Linux е като използвате твърд диск. Linux поддържа всички MFM, RLL и IDE контролери. Освен това се поддържат повечето ESDI контролери (тези, които имат хардуерна емулация на стандарта ST506).

Общото правило за твърдите дискове и дискетните устройства, които не са със SCSI интерфейс е, че ако можете да използвате това устройство под Windows или друга операционна система, би трябвало да можете да го използвате и под Linux.

Освен това, Linux поддържа повечето популярни SCSI контролери, макар че поддръжката за SCSI е по-ограничена, поради голямото количество интерфейсни стандарти. Поддържат се SCSI контролерите Adaptec AHA2940, AHA3940, AHA1542B, AHA1542C, AHA1742A (BIOS v1.34), AHA1522, AHA1740 (SCSI-2 контролер, BIOS v1.34 в Enhanced mode); Future Domain 1680, TMC-850, TMC-950; Seagate ST-02; UltraStor SCSI; Western Digital WD7000FASST. Би трябвало да можете да използвате и контролери на други фирми, базирани на тези карти. Този списък е само кратка извадка от поддържаните контролери. Пълният списък е твърде голям, за да бъде включен в книгата.

Изисквания за пространството на твърдия диск

Разбира се, за да инсталирате Linux, трябва да отделите известно количество свободно пространство на твърдия си диск. Linux може да работи

с няколко твърди диска в един компютър; ако е необходимо, можете да отделите пространство за Linux на няколко диска.

Размерът на необходимото ви дисково пространство зависи предимно от нуждите ви и от софтуера, който ще инсталирате. Linux е сравнително малък - за да инсталирате пълна система, са ви необходими от 10 до 20 MB. Все пак, ако искате да имате място за разширяване на системата или за по-големи пакети като системата X Window, трябва да отделите повече дисково пространство. Ако планирате системата да се използва от много потребители, трябва да отделите място за съхранение на техните файлове.

Освен това, почти сигурно е, че ще искате да отделите място на твърдия диск за виртуална памет. Ще разгледаме детайлите за инсталирането и използването на виртуална памет в раздела "Управление на виртуалната памет" в Глава 6, *Управление на файловите системи, виртуалната памет и файловете на хардуерните устройства.*

Глава 6

Всяка дистрибуция на Linux се разпространява с документация, която би трябвало да ви помогне да прецените количеството дисково пространство, от което се нуждаете, в зависимост от количеството софтуер, което планирате да инсталирате. За Linux система с минимални възможности са ви необходими по-малко от 20 MB; за пълна система с всички удобства - около 300 MB; за много голяма система с място за файловете на много потребители и свободно пространство за бъдещи разширения трябва да отделите около 1 GB. Разбира се, това са приблизителни оценки; трябва да прецените собствените си нужди и цели, за да определите количеството дисково пространство, което ще задоволи конкретните ви изисквания.

Изисквания за монитор и видео-контролер

Linux поддържа всички стандартни видео-контролери (Hercules, CGA, EGA, VGA, IBM Monochrome и SuperVGA) и монитори за стандартния си текстов интерфейс. Ако видео-контролерът и мониторът на вашия компютър работят под друга операционна система (например под Windows), би трябвало да работят добре и под Linux. Оригиналните видео-контролери IBM CGA страдат от така наречения "ефект на снежинките" под Linux, което е неприятно (ако имате такава платка, ви съветваме да я подарите на историческия музей).

Ш Графичните среди като системата X Window имат свои собствени изисквания за видео хардуера. Вместо да ги описваме тук, ще отложим дискусията до раздела "Хардуерни изисквания" в Глава 10. Накратко, за да

Глава ю

използвате системата X Window на вашия компютър, ще ви бъде необходим някой от видео-контролерите, описани в посочения раздел. Добрата новина е, че се поддържат почти всички видео-карти (включително и най-съвременните графични ускорители).

Друг хардуер

В предишните раздели описахме хардуера, който е необходим, за да можете да използвате Linux. Все пак, повечето потребители имат някои "допълнителни" устройства, като ленти, CD-ROM, звукови платки и т.н. и се интересуват дали техният хардуер се поддържа от Linux.

Мишка и други посочващи устройства

В повечето случаи ще използвате мишката само при работа в графични среди като системата X Window. Разбира се, съществуват и приложения с текстов интерфейс, които също могат да използват мишка.

Linux поддържа всички стандартни мишки със сериен интерфейс, включително Logitech, сериите MM, Mouseman, Microsoft (с два бутона) и Mouse Systems (с три бутона). Освен това, Linux поддържа мишки с интерфейс PS/2 и busmouse на Microsoft, Logitech и ATIXL.

Също така би трябвало да работят и всички други посочващи устройства, които емулират изброените по-горе мишки (например trackball). .

CD-ROM и DVD-ROM

Много от съвременните CD-ROM устройства използват почти универсалния интерфейс IDE/ATAPI, който изцяло се поддържа от Linux. Други CD-ROM устройства използват интерфейса SCSI и ако Linux поддържа SCSI-контролера ви, ще можете да използвате и SCSI CD-ROM устройство си под Linux.

Linux поддържа стандартната за компакт-дискове файлова система ISO-9660, включително разширението на Microsoft за дълги имена.

Освен CD-ROM устройствата, Linux поддържа и някои CD-R и CD-RW устройства, които можете да използвате, за да създавате собствени компакт-дискове.

Linux-ядрото с версия 2.2 поддържа разнообразни DVD-ROM устройства, но все още не е включена файлова система, която да позволява директен достъп до съдържанието на DVD. Предполагаме, че по времето, по което четете тази книга, в ядрото ще има пълна поддръжка на DVD.

Лентови и други устройства със сменяеми носители

На пазара се предлагат различни типове лентови устройства. Много от тях използват интерфейса SCSI и би трябвало се поддържат изцяло под Linux. Освен това се поддържат QIC-02 и така наречените "floppy tape" устройства (които се включват към контролера на дискетното устройство), а също и различни типове устройства със сменяем носител, като DAT и Iomega ZIP.

Принтери

Linux поддържа цялостна гама паралелни принтери. Ако можете да използвате принтера си през паралелния порт от Windows или друга операционна система, би трябвало да можете да го използвате и от Linux. Софтуерът за печат под Linux се състои от стандартните за Unix пакети *lp* **и** *lpr*. **Този софтуер ви позволява да печатате отдалечено (през мрежа).**

Модеми

Linux поддържа пълна гама серийни модеми (както вътрешни, така и външни). Ако модемът ви може да се използва от друга операционна система на същия компютър, най-вероятно няма да имате трудности да го използвате под Linux. Все пак, Linux не поддържа така наречените "Winmodem"-и - лишени от елегантност устройства, при които операционната система трябва да се грижи за много от функциите на модема. Освен това не се поддържат и някои от вътрешните PCI-модеми.

Ethernet, Fast Ethernet и Gigabit Ethernet контролери

Linux поддържа почти всеки Ethernet или Fast Ethernet контролер за PC платформа, а за по-голямата част от останалите се разработват драйвери. Ядрото с версия 2.2 включва драйвери за някои високопроизводителни мрежови карти, включително Packet Engines G-NIC, Alteon AceNIC и 3Com 3C985 PCI Gigabit Ethernet. Поддържат се и много token-ring и ATM контролери, както и различни System Area Networks карти като Myricom Myrinet.

Източници на информация за Linux

Както вероятно сте се досетили, освен тази книга съществуват още много други източници на информация за Linux.

Електронни документи

Ако имате достъп до Интернет, можете да изтеглите много документи за Linux от web и от т.нар. "анонимни" FTP-сайтове. Дори да нямате достъп до Интернет, можете да намерите част от тези документи на компакт-диск - много дистрибуции на Linux съдържат всички документи, които споменаваме тук. Освен това, можете да откриете документация за Linux в много други мрежи като Fidonet и CompuServe.

Съществуват много web и FTP-сайтове, които съдържат софтуер за Linux със съответната документация. Приложение А съдържа списък на някои от документите за Linux, които можете да изтеглите от Интернет.

Примери за съществуваща електронна документация са Linux FAQ (Frequently Asked Questions) - колекция от най-често задаваните въпроси за Linux (разбира се, заедно с отговорите им); документите Linux HOWTO (HOWTO буквално означава "как да ...") - всеки от тях описва специфичен аспект на системата, например Installation HOWTO описва как се инсталира Linux, Printing HOWTO описва как се печата под Linux, Ethernet HOWTO описва как можете да използвате с Ethernet контролери в Linux и накрая Linux META-FAQ - списък с други източници на информация за Linux в Интернет.

Повечето от тези документи се изпращат редовно на една или повече от свързаните с Linux групи в Usenet; виж раздела "Групи по интереси в Usenet" по-долу в тази глава.

Интернет-сайтът на Linux Documentation Project (LDP - проект за документиране на Linux) на адрес *http://www.linuxdoc.org* съдържа повечето документи HOWTO, справочници от LDP (виж следващия раздел) и много други документи, както и препратки към сайтове, полезни на потребителите на Linux.

Книги и други публикации

Библиографията в края на тази книга съдържа препратки към огромно количество източници на информация, които могат да ви помогнат да използвате системата. Съществуват множество публикации специално за Linux. Най-забележителни са книгите от Linux Documentation Project (LDP - проект за документиране на Linux). Целта на LDP е да се създадат и разпространят истински "ръководства" за Linux с високо качество. Тези ръководства са аналогични на документациите, които се предоставят с платените версии на Unix и обхващат всичко за Linux от инстали-

рането до използването и поддръжката на системата, програмирането, работата в мрежа, разработката на ядрото и т.н.

Библио-
графия

Ръководствата на LDP могат да бъдат изтеглени от Web или закупени по пощата от различни източници. В Библиографията са описани детайлно готовите ръководства и начините за получаването им. Издателството O'Reilly & Associates вече публикува книгата *Linux Network Administrator's Guide* от LDP.

Освен нарастващия брой книги за Linux, съществува голямо количество книги за Unix изобщо, които са директно приложими за Linux - що се отнася до използването и програмирането, в повечето отношения Linux не се различава много от другите реализации на Unix. Всъщност, тази книга е написана като допълнение към огромната библиотека вече написани книги за Unix и в нея се описват най-важните, специфични за Linux детайли. Авторите се надяват да използвате и други източници за по-подробна информация.

Въоръжени с няколко добри книги за Unix, както и с тази книга в ръцете си, би трябвало да сте в състояние да се справите с почти всеки проблем. Библиографията съдържа списък с книги за Unix (които горещо ви препоръчваме), полезни както за начинаещи, така и за експерти.

Съществуват поне две месечни списания за Linux - *Linux Journal* и *Linux Magazine.* Те представляват чудесен начин да сте в течение с много от тенденциите и текущите разработки в Linux-обществото.

Групи по интереси в Usenet

Usenet е световен форум за електронни новини и дискусии с голям брой от така наречените "групи по интереси" (newsgroups) - място за дискусия, посветено на определена тема. Голяма част от разработката на Linux е извършена с използването на Internet и Usenet и затова не е изненадващо, че съществуват много групи в Usenet, посветени на дискусии за Linux.

Интернет услугата mailing list

Дори да нямате достъп до Usenet, ако имате достъп до електронна поща в Интернет, можете да се включите в редица списъци, наречени mailing lists. Mailing list е списък на потребители, желаещи да получават новини за конкретна тема. Дори да нямате директен достъп до Интернет, можете да се присъедините към някой mailing list, стига да имате начин да

разменяте електронна поща с Интернет (някои мрежи като UUCP, Fidonet, CompuServe и други имат достъп до Интернет-поща).

Как да получим помощ

Без съмнение ще се нуждаете от известна помощ по време на приключенията си в света на Linux. Дори най-опитните потребители на Unix понякога се затрудняват от някои особености на Linux. Затова е важно знаете къде и как да потърсите помощ при необходимост.

Основният начин за получаване на помощ в света на Linux са Internet mailing lists и групите в Usenet. Ако нямате директен достъп до тези източници, може би ще успеете да намерите подобно място за дискусии, посветени на Linux, в други електронни източници, като локални BBS-и, CompuServe и т.н.

Някои компании предлагат платена поддръжка на Linux. Заплащането на определена "абонаментна такса" ви позволява да се консултирате по телефона и да получавате помощ при решаването на проблемите си. Някои дистрибутори на Linux също предлагат платена поддръжка. А ако имате достъп до Usenet и email, ще откриете колко полезна може да бъде дори безплатната поддръжка на Linux.

Ще придобиете по-голям опит с Linux и ще имате по-голям успех при търсенето на помощ за разрешаването на проблемите си, ако спазвате следните препоръки:

Първо прочетете цялата налична документация.

Прилож.А

Първото нещо, което трябва да направите, когато се сблъскате с проблем, е да се консултирате с различните източници на информация, цитирани в предишния раздел и в Приложение А. Тези документи са грижливо написани за хора като вас - хора, които се нуждаят от помощ при работата си с Linux. Дори общите книги за Unix са приложими за Linux и би трябвало да се възползвате и от тях. Почти сигурно е, че ще намерите решението на вашия проблем някъде в тази документация.

Ако имате достъп до Web, Usenet или свързан с Linux mailing list, прочетете внимателно намиращата се в тях информация, преди да изпратите заявка за помощ. Не винаги е лесно в документацията да се намери решение на често срещаните проблеми, докато в посветените на Linux групи в Usenet и mailing lists на тези проблеми се обръща особено внимание. Ако не четете внимателно информацията в

тези форуми, а само изпращате заявки за помощ, вероятно ще си създадете проблеми.

Ако не можете сами да откриете решение на проблема, използвайте машините, предназначени за търсене в web (т.нар. search engines), или архивите на Usenet в DejaNews на следния адрес:

http://www. dejanews. com

Научете се да цените самостоятелното решаване на проблемите.

В повечето случаи за предпочитане е да изследвате самостоятелно колкото е възможно по-задълбочено проблема, преди да потърсите външна помощ. Linux насърчава самостоятелното решаване (хакването) на проблемите. Той не е платена операционна система и не се и опитва да изглежда като такава. Хакването няма да ви убие. Всъщност, то ще ви научи на много неща за системата, така че да можете сами да изследвате и решавате евентуалните проблеми. Възможно е дори да придобиете умения, достатъчни, за да ви наричат Linux-гуру. Научете се да цените хакването на системата и самостоятелното решаване на проблемите. Не би трябвало да очаквате да използвате пълноценен, създаден от вас Linux без известно количество "ръчна" работа.

Запазете спокойствие.

Жизнено важно е да не бързате да се разочаровате от системата. Няма да спечелите нищо, ако в пристъп на гняв приложите брадва (или мощен електромагнит) към вашата Linux система. В случай на стрес бихте могли чудесно да се облекчите, ако удряте с юмрук голяма възглавница или друг неодушевен предмет. Надяваме се, че със създаването на Linux, дистрибуциите ще станат по-стабилни и повечето проблеми ще изчезнат. Дори платените реализации на Unix понякога създават проблеми. Ако нищо не помага, отпуснете се, поемете няколко пъти дълбоко въздух и след като си починете се заемете отново с проблема. Умът ви ще бъде по-ясен и системата ще ви бъде благодарна.

Не бързайте да търсите помощ

Много хора допускат грешката преждевременно да изпратят съобщение за помощ. Когато имате проблем, не бързайте, повтаряме, *не бързайте* да се хвърлите към най-близкия терминал и да изпратите съобщение до някоя от свързаните с Linux групи в Usenet. Често след 5 минути откривате, че сте допуснали елементарна грешка и се оказвате в неприятна ситуация, в която се налага да защитавате здравия си разум в публичен форум. Преди да изпратите каквото и

да е до някоя Linux-група, се опитайте сами да разрешите проблема или поне се уверете какъв точно е той. Например, ако компютърът ви не тръгва, проверете дали не е изключен от контакта.

Ако търсите помощ, направете го разумно.

Ако всичко друго пропадне, бихте могли да изпратите заявка за помощ в посветените на Linux електронни форуми, като някои от групите в Usenet и mailing lists. Не забравяйте, че хората, които четат вашето съобщение, не са във форума, за да ви помогнат. Мрежата не е вашия личен консултант. Затова е важно да бъдете колкото е възможно по-учтиви, ясни и да предоставите изчерпателна информация за проблема си.

Как се постига това? Първо, трябва да включите колкото е възможно повече (уместна) информация за вашата система и за проблема, който имате. Няма да постигнете много, ако изпратите съобщение от типа на "Не мога да накарам електронната поща да работи". Би трябвало да включите информация за системата си, какъв софтуер използвате, какво сте се опитали да направите, за да решите проблема, и какви са резултатите. Когато давате техническа информация, обикновено трябва да споменете коя е версията на използвания от вас софтуер (например на ядрото на Linux), както и да опишете накратко каква е хардуерната ви конфигурация. Не прекалявайте - обикновено не е необходимо да описвате подробно какъв монитор използвате, ако се опитвате да конфигурирате мрежов софтуер.

Второ, преди да използвате Мрежата, би трябвало да направите опит - колкото и да е незначителен - сами да решите проблема. Ще допуснете голяма грешка, ако (например) никога не сте се опитвали да настроите електронната поща и веднага решите да попитате момчетата в Мрежата как става това. Съществуват множество документи (виж предишния раздел "Източници на информация за Linux"), в които е описано как се решават основните задачи под Linux. Идеята е да стигнете колкото е възможно по-далече сами и чак *тогава* (ако се наложи) да потърсите помощ.

Не забравяйте, че хората, които четат вашето съобщение, дори да имат желание да помогнат, понякога се дразнят, когато отново и отново виждат съобщения за един и същи проблем. Прочетете внимателно документите HOWTO, FAQ, архивите на групите в Usenet и mailing lists, преди да изпратите съобщение за вашия проблем. Често решението му е описано многократно и единственото, което трябва да направите, е да прочетете готовите документи.

Трето, когато изпращате съобщение до някоя група в Usenet или mailing list, опитайте се да бъдете колкото е възможно по-учтиви, директни и информативни. Много по-ефективно и разумно е да бъдете учтиви и откровени - повече хора биха искали да ви помогнат, ако сте по-скромни. Не се въвличайте в словесна война, защото това води до загуба на време - вашето и на други хора. Електронната комуникация е чудесен начин да получите помощ при решаването на проблемите си, но е важно да знаете как да използвате Мрежата *ефективно*.

ГЛАВА 2

ПОДГОТОВКА ЗА ИНСТАЛИРАНЕ НА LINUX

В тази глава ще ви запознаем с подготовката за инсталиране на Linux. Ще опишем как можете да се снабдите с предварително подготвени дистрибуции на Linux и как да подготвите компютъра си за инсталиране. Ще опишем няколко начина, по които да разделите твърдия си диск на логически дялове, така че да е възможно инсталирането на Linux на един и същ диск с Windows, OS/2 или друга операционна система.

Както вече споменахме, не съществува "официална" дистрибуция на софтуера за Linux. Съществуват много дистрибуции, всяка от които е подходяща за конкретни цели. Тези дистрибуции могат да бъдат изтеглени чрез анонимен FTP достъп от Интернет, от BBS системи или да се закупят на дискета, лента или компакт-диск.

Дистрибуции на Linux

Понеже Linux е свободен софтуер, не съществува единна организация, която да се грижи за създаването и разпространяването му. Всеки, който има желание, може да подбере и разпространява софтуер за Linux като спазва ограниченията на GPL. Затова съществуват много дистрибуции на Linux, които могат да се изтеглят от Интернет или да се закупят от съответните доставчици.

Пред вас стои задачата да изберете такава дистрибуция на Linux, която отговаря на нуждите ви. Между отделните дистрибуции съществуват разлики. Много от тях се разпространяват с целия софтуер, който е необходим за една завършена система. Други дистрибуции са предназначени за потребители, които нямат голямо количество свободно дисково

Глава 5

пространство. Много дистрибуции съдържат само основния софтуер за Linux - очаква се потребителят сам да инсталира по-големите софтуерни пакети, например системата X Window (виж Глава 5, *Основи на системната администрация).*

Документът Linux Distributions HOWTO съдържа списък с дистрибуциите на Linux, които могат да се получат от Интернет или по пощата.

Как да изберете коя е подходящата за вас дистрибуция? Ако имате достъп до Usenet или друг електронен форум, можете да попитате участниците в него, които са инсталирали Linux, какво е мнението им за различните дистрибуции. Още по-добре би било, ако имате познат, който е инсталирал Linux - можете да го помолите за съвет и помощ. В действителност повечето дистрибуции на Linux съдържат практически един и същ софтуер, така че изборът, който правите, е повече или по-малко случаен.

Как да закупим Linux

Ако нямате достъп до Интернет или BBS система, след съответното заплащане можете да получите по пощата голяма част от дистрибуциите на Linux - на дискета, лента или компакт-диск. Повечето дистрибутори приемат плащания с кредитни карти и платежни нареждания, така че където и да живеете, бихте могли да получите Linux по този начин.

Linux е безплатен софтуер, но GPL позволява на дистрибуторите да печелят от продажбата му. Затова в зависимост от дистрибуцията, цената варира между 5 и 150 щатски долара. Но ако познавате някой, който вече е закупил или изтеглил от Интернет дистрибуция на Linux, можете свободно да я заемете или копирате за ваша собствена употреба. Дистрибуторите на Linux по никакъв начин не могат да ограничават лиценза или разпространението на софтуера. Например, ако искате да инсталирате Linux в цяла лаборатория с компютри, достатъчно е да закупите едно единствено копие на някоя от дистрибуциите, което можете да използвате за инсталиране на Linux на всяка от машините. Все пак съществува едно изключение от това правило - някои компании добавят към дистрибуцията платени софтуерни пакети, които не могат да бъдат инсталирани на няколко компютъра. В този случай, това трябва да е изрично отбелязано на пакета.

Много потребителски групи предлагат собствени дистрибуции на Linux - може би ще намерите такава група близо до вас. Потребителските групи са особено полезни, ако използвате специални платформи, например Alpha.

Как да изтеглим Linux от Интернет

Ако имате достъп до Интернет, най-лесният начин да се сдобиете с Linux е като използвате анонимен FTP достъп. Един от основните FTP-сайтове е *ftp://metalab.unc.edu* - можете да намерите различни дистрибуции на Linux в директорията */pub/Linux/distributions.*

Много от дистрибуциите могат да се изтеглят от анонимни FTP сайтове във вид на "копия" на дискети (disk image). Това означава, че дистрибуцията се състои от комплект файлове, всеки от които съдържа двоично копие на данните в една дискета. За да прехвърлите данните от файла върху дискета, можете да използвате програмата *RAWPJTE.EXE,* която работи под MS-DOS. Тази програма копира блок по блок съдържанието на файла върху дискета, независимо от формата на данните. Можете да изтеглите програмата *RA WRITE.EXE* от различни FTP сайтове със софтуер за Linux, например от сайта *ftp://metalab.unc.edu* (виж директорията */pub/Linux/system/Install/rawrite).*

Предупреждаваме ви, че този начин на инсталиране на Linux изисква много труд - дистрибуцията може да се състои от повече от 50 дискети.

И така, в много случаи можете просто да изтеглите комплект файлове от Интернет и като използвате *RA WRITE.EXE* да създадете от тях дискети, от които да инсталирате Linux. Трябва да заредите операционна система от една от дискетите, наречена "boot fIoppy" (т.е. дискета за зареждане), след което можете да започнете. Обикновено софтуерът се инсталира директно от дискетите, макар че някои дистрибуции могат да се инсталират от дял на MS-DOS на твърдия диск. Някои дистрибуции могат да се инсталират през TCPJIP мрежа. Методите, по които може да се инсталира дадена дистрибуция, обикновено са описани в документацията й.

Други дистрибуции на Linux се инсталират от множество дискети в MS-DOS формат. Например, Linux дистрибуцията на Slackware изисква *RA WRITE.EXE* само за две дискети - boot и root. Останалите дискети са във формата на MS-DOS и се създават като се използва MS-DOS командата COPY. Системата инсталира софтуера директно от тези дискети. Това ви спестява неприятностите при използването на *RAWRITE.EXE* с голямо количество "копия" на дискети, но изисква MS-DOS, за да създадете дискетите.

Ако имате достъп до Unix работна станция с дискетно устройство, можете да използвате командата *dd,* за да запишете данните от image-файл

* Ако нямате директен достъп до Интернет, но можете да изпращате и получавате електронна поща, можете да получите Linux в пощенската си кутия, като използвате услугата FTPMAIL.

директно върху дискета. Ако използвате работна станция на Sun, командата `dd of=/dev/rfd0 if=foo` bs=18k ще запише "директно" съдържанието на файла ***foo*** върху дискета. Повече информация за дискетните устройства на системата ви и начина на използване на командата *dd* можете да получите от някой опитен Unix потребител.

Всяка дистрибуция на Linux, която може да се получи чрез анонимен FTP достъп, би трябвало да съдържа файл с име *README* ("прочети ме"), в който е описано как да изтеглите и да подготвите дискетите за инсталация. Прочетете внимателно цялата налична документация за дистрибуцията, която смятате да инсталирате.

Когато изтегляте софтуер за Linux от Интернет, трябва задължително да използвате двоичен режим при прехвърлянето на всички файлове (при повечето FTP-клиенти можете да използвате командата *binary,* за да включите този режим).

Как да изтеглим Linux от други електронни източници

Ако имате достъп до BBS система, може би ще успеете да изтеглите от нея Linux. Не всяка дистрибуция на Linux, може да бъде получена по този начин (особено дистрибуциите на компакт-диск - тях обикновено трябва да закупите).

Подготовка за инсталиране на Linux

След като вече имате дистрибуция, можете да започнете подготовката на системата си за инсталирането на Linux. Необходимо е известно планиране, особено ако вече използвате друга операционна система. В следващите раздели ще опишем как да планирате инсталирането на Linux.

Инсталирането накратко

Макар че всяка дистрибуция е различна, най-общо процедурата за инсталиране на Linux е следната:

1. *Отделете дисково пространство за Linux.* Ако вече имате инсталирана друга операционна система, трябва да преразпределите пространството в твърдите си дискове, за да отделите място за Linux (виж раздела "Преразпределяне на твърдия диск" по-долу в тази глава). В някои дистрибуции тази стъпка е част от инсталационния процес. За да проверите дали това е така и за дистрибуцията, която използвате, прочетете документацията й. Все пак, дори ако вашата дистрибуция може да извърши тази стъпка, няма да се случи нищо

лошо, ако следвате нашите инструкции и предварително отделите дисково пространство за Linux.

2. *Заредете Linux от инсталационния носител.* Всяка дистрибуция на Linux има някакъв вид инсталационен носител - обикновено това е дискета или компакт-диск, от който може да се зареди операционна система (bootable CD-ROM). Най-често след зареждането на Linux автоматично тръгва някакъв вид инсталираща програма, която ще ви ръководи по време на инсталирането. При някои дистрибуции имате възможност да инсталирате софтуера "ръчно".

3. *Създайте дялове за Linux.* След като сте отделили дисково пространство, в него трябва да създадете дялове (partitions) за Linux. Това обикновено се прави с програмата *fdisk* за Linux (виж раздела "Създаване на дялове за Linux" в Глава 3. *Инсталиране и начално конфигуриране)* или с някоя друга програма, специфична за дистрибуцията, която използвате (например *Disk Druid* от Red Hat Linux).

4. *Създайте (файлови системи и място за swap.* На този етап в дяловете на Linux се създават една или повече файлови системи, които се използват за съхранение на файлове. Ако смятате да използвате виртуална памет, в този момент трябва да създадете място за swap в един от дяловете на Linux. Тези стъпки са описани в разделите "Създаване на файлови системи" и "Създаване на място за swap" в Глава 3.

5. *Инсталирайте софтуера в новите файлови системи.* Накрая трябва да инсталирате софтуера за Linux в новите файлови системи (виж предишната стъпка). Ако всичко върви добре, след тази стъпка ще имате работеща инсталация на Linux. Процесът е описан в раздела "Инсталиране на софтуера" в Глава 3. Раздела "Какво да правим, ако възникнат проблеми" в същата глава съдържа съвети, които ще ви помогнат за разрешаване на евентуалните проблеми.

Понякога хората, които искат да използват няколко различни операционни системи на един компютър, се питат коя от тях да инсталират първо - Linux или другата операционна система. Обикновено хората имат проблеми, когато инсталират Windows 95 след Linux. При инсталирането си Windows изтрива съществуващата информация за начално зареждане, затова е по-добре да инсталирате първо Windows, а след това, като използвате информацията в тази глава, да инсталирате Linux.

Не знаем дали Windows 98 ще демонстрира същата безцеремонност като Windows 95 .

* Демонстрира я - б.пр.

Windows NT изглежда е по-толерантен към съществуващата информация за начално зареждане.

Много дистрибуции на Linux предлагат инсталираща програма, която ще ви ръководи по време на инсталационния процес и ще извърши автоматично някои от изброените стъпки. Докато четете тази и следващата глава, не забравяйте, че в зависимост от дистрибуцията, която използвате, е възможно някои от описаните процедури да бъдат извършени автоматично.

Важно! Най-добрият съвет, който можем да ви дадем, е да си водите бележки по време на цялата процедура по инсталирането на Linux. Записвайте в бележника си всичко, което правите, всеки текст, който въвеждате и всяко съобщение, което ви се струва необичайно. Идеята е проста - ако (или когато!) възникне проблем, ще можете да проследите стъпките си и да откриете какво е предизвикало проблема. Инсталирането на Linux не е трудно, но има много детайли, които трябва да се запомнят. Хубаво е да имате описание на тези детайли, за да можете да опитате други начини на инсталиране, ако нещата не вървят както трябва. Освен това, вашите бележки ще помогнат, ако потърсите съвет от други хора - например, ако изпратите съобщение до някоя свързана с Linux група в Usenet. И накрая, бележките ви ще са нещо, което бихте искали някой ден да покажете на внуците си.*

Основни понятия при разделянето на твърдия диск на дялове

Най-общо казано, твърдите дискове се разделят на *дялове* (partitions), като за всяка операционна система се отделят един или няколко дяла. Например, на един твърд диск можете да имате няколко отделни дяла - да кажем един за Windows, друг за OS/2 и още два за Linux.

Ако вече имате инсталиран софтуер на компютъра си, може да се наложи да промените размера на дяловете в твърдия диск, за да освободите място за Linux. След това в освободеното пространство трябва да създадете един или повече дяла за Linux, в които по-късно ще запишете софтуера и ще използвате за виртуална памет. Ще наричаме този процес *преразпределяне* на твърдия диск.

Много MS-DOS системи използват един единствен дял, който заема целия твърд диск. В MS-DOS този дял е известен като C:. Ако имате пове-

* Мат Уелш признава, че е записвал всичките си премеждия с Linux в първите няколко месеца, през които е работил със системата. В момента бележките му събират прах в библиотеката.

ват всичките си файлове в основната файлова система, което в много случаи е по-лесно, отколкото поддръжката на няколко файлови системи и съответно няколко дяла на твърдия диск.

Разбира се, ако желаете, можете да създадете няколко файлови системи за Linux - например, ако искате да имате отделни файлови системи за директориите ***/usr*** и ***/home.*** Читателите с опит в системната администрация на Unix знаят как да използват много файлови системи по подходящ начин. Ще ви покажем как можете да използвате няколко файлови системи в раздела "Създаване на файлови системи" в Глава **6.**

Глава 6

Защо е необходимо да се използва повече от една файлова система? Основната причина е сигурността. Ако поради някаква причина една от вашите файлови системи се повреди, обикновено останалите остават невредими. От друга страна, ако използвате една единствена файлова система за всичките си файлове и по някаква причина тя се повреди, възможно е да загубите всичките си данни наведнъж. Все пак, това се случва доста рядко - ако редовно архивирате данните си, би трябвало да нямате сериозни проблеми.

Освен това, използването на няколко файлови системи позволява лесно да осъвременявате софтуера, без да рискувате данните си. Ако имате отделен дял за личните директории на потребителите, когато актуализирате системата си, можете спокойно да изтриете всички други дялове и да направите изцяло нова инсталация на Linux. Разбира се, всички съвременни дистрибуции на Linux имат доста добри процедури за актуализация, но е възможно просто да решите да започнете "на чисто".

Друга причина да използвате много файлови системи е разпределянето на файловете в няколко твърди диска. Например, ако имате два твърди диска със свободно дисково пространство съответно 100 MB в единия и 2 GB в другия, бихте могли да използвате първия диск за основната файлова система, а втория - за файловата система ***/usr.*** За момента не е възможно една файлова система да се разположи на няколко диска, затова се налага да използвате няколко файлови системи, когато свободното ви пространство е разпределено върху няколко твърди диска.

Накратко, Linux изисква най-малко един дял от диска за основната файлова система. Ако искате да използвате много файлови системи, трябва да отделите дял за всяка допълнителна файлова система. Някои дистрибуции на Linux автоматично създават задължителните дялове и файлови системи, така че може изобщо да не се наложи да се грижите за тези въпроси.

*

Мат Уелш използва една 200 мегабайтова файлова система за всичките си свързани с Linux файлове и досега не е имал никакви проблеми.

Когато планирате преразпределянето на твърдия си диск, трябва да вземете под внимание и мястото, необходимо за swap. Имате две възможности. Първата е да използвате *swap-файл* (буквално "файл за замяна"), който ще се намира в една от файловите системи на Linux. Този файл може да бъде създаден след като инсталирате софтуера. Втората възможност е да създадете *дял за swap* (буквално "дял за замяна"). Това е отделен дял, който ще се използва единствено за виртуална памет. Повечето хора предпочитат да използват втората възможност.

Максималният размер на файл или дял за swap е 128 MB (при последните версии на ядрото максималният размер е по-голям).* Ако искате да използвате повече от 128 MB swap (макар че едва ли ще ви се наложи), трябва да създадете няколко файла или дяла за swap - общо до 16 на брой. Например, ако искате да имате 256 MB swap, можете да създадете два дяла с размер по 128 MB.

Създаването на дял за swap е описано в раздела "Създаване на място за swap" в Глава 3, а създаването на swap-файл - в раздела "Управление на виртуалната памет" в Глава 6.

Следователно, общото правило е следното: трябва да имате поне два дяла за Linux - един за основната файлова система и друг за swap. Това са минималните изисквания, като разбира се, съществуват много други варианти за разделяне на твърдия диск. Linux не ви задължава да използвате виртуална памет, но горещо ви го препоръчваме, ако имате помалко от 32 MB физическа памет.

Разбира се, трябва да знаете колко *дисково пространство* ви е необходимо. Размерът на файловите системи, които съдържат софтуера за Linux, зависи преди всичко от това какви пакети смятате да инсталирате и коя дистрибуция използвате. Документацията, която е включена в дистрибуцията, може да ви даде приблизителна представа за необходимото дисково пространство. За една малка Linux система са достатъчни 40 MB; по-големите системи изискват от 100 до 300 MB или повече. Не забравяйте, че освен за софтуера, трябва да отделите място и за потребителските директории, за бъдещо разрастване на системата и т.н.

Ако използвате няколко дяла, можете да отделите сравнително малък дял за основната (/) директория - 32 MB би трябвало да са достатъчни. Добавете поне още 30-50 MB, ако решите да оставите директорията */var* на същия дял, както правят повечето хора. От друга страна, най-вероятно бихте искали да използвате значително по-голям дял за */usr*.

* Тази стойност е валидна за машини с процесор Intel. Тя може да е по-голяма за някои други архитектури, например за Alpha.

Размерът на дяла за swap (ако решите да го използвате) зависи от количеството виртуална памет, от което се нуждаете. Емпиричното правило е да използвате място за swap с размер два пъти по-голям от размера на физическата памет, която имате. Например, ако имате 4 MB физическа памет, би трябвало да е достатъчен swap-дял с размер 8 MB. Разбира се, това е чисто предположение. Действителното количество място за swap, което ви в необходимо, зависи от софтуера, който ще използвате. Ако имате голямо количество физическа памет (например 64 MB или повече), може да решите изобщо да не използвате swap.

Заради ограничения на BIOS, обикновено не е възможно да заредите операционна система от дял, който започва след 1023-я цилиндър на диска. Затова, когато отделяте място за Linux, не забравяйте, че е възможно да имате проблеми, ако дялът с основната файлова система на Linux е след 1023-я цилиндър. Linux може да *използва* дялове, които се намират след този цилиндър, но може би няма да можете да *заредите* Linux от такъв дял. Този съвет може би изглежда преждевременен, но е важно да го имате в предвид, когато разпределяте дисковото си пространство. Още повече, днес се използват големи дискове с много повече от 1023 цилиндъра.

Ако за основната файлова система на Linux е абсолютно наложително да използвате дял, който започва след 1023-я цилиндър, винаги можете да заредите Linux от дискета. Всъщност това не е чак толкова лошо; необходими са само няколко секунди повече, отколкото при зареждане от твърд диск. При всички случаи, винаги можете да използвате тази възможност.

Преразпределяне на твърдия диск

В този раздел ще опишем как да промените размера на съществуващите дялове на твърдия диск (ако имате такива), за да освободите място за Linux. Ако инсталирате Linux на празен диск, можете да прескочите този раздел и да продължите с Глава 3.

Обикновено, когато искате да промените размера на съществуващ дял, просто трябва да го изтриете (като по този начин унищожавате всички данни в него) и да го създадете отново. Преди да преразпределите пространството в твърдия си диск, *архивирайте програмите и данните в системата си.* След като промените размера на дяла, можете да възстановите оригиналния софтуер от архивното копие. Все пак съществуват няколко програми за MS-DOS, с които можете да промените размера на съществуващите дялове, без да ги унищожавате. Една от тези програми се нарича FIPS и може да се изтегли от много сайтове със софтуер за Linux.

Не забравяйте, че е възможно заради намаляването на размера на дяловете, мястото в тях да не е достатъчно за инсталирането на целия предишен софтуер. В този случай трябва да премахнете излишния софтуер, за да може останалия да се събере в тези по-малки дялове.

Програмата, която се използва за преразпределяне на твърдия диск, е известна ***txtofdisk.*** Всяка операционна система има свой аналог на тази програма. Например, под MS-DOS можете да използвате командата FDISK. Консултирайте се с документацията на операционната система, която използвате в момента, за да научите как се извършва преразпределянето на диска. Тук ще опишем само как да извършите тази процедура в MS-DOS като използвате *fdisk,* но тази информация би трябвало лесно да се приложи и за други операционни системи.

Програмата *fdisk* (на всяка операционна система) се използва за прочитане на таблицата с дялове на определен диск и при нужда за променянето ѝ като се добавят или премахват дялове. Някои версии на *fdisk* могат да изпълняват и други задачи, например да добавят информация в началото на нов дял, така че да го подготвят за използване от определена операционна система. За нашите цели е достатъчно *fdisk* да създаде дялове за операционната система, на която принадлежи. Не можете да създадете дял за MS-DOS като използвате *fdisk* от Linux; дяловете, които се създават по този начин, не могат да бъдат използвани коректно от MS-DOS (всъщност, ако наистина знаете какво правите, можете да създадете дялове за MS-DOS като използвате Linux, но не ви съветваме да го правите), а *fdisk* на MS-DOS не може да разпознае дялове на Linux. Все пак, не би трябвало да имате проблеми, ако създавате дялове за определена операционна система със съответстващия ѝ *fdisk*. (Забележка: В някои операционни системи тази програма има друго име, например "disk manager" или "volume manager".)

В раздела "Създаване на дялове за Linux" в Глава 3 описваме как се създават дялове за Linux. В този раздел обясняваме само как се променя размера на съществуващите дялове.

Моля, прочетете документацията на операционната система, която използвате, преди да преразпределите пространството на твърдия си диск. В този раздел е дадено общото описание на процеса, но съществуват много тънкости, които не са описани тук. Възможно е да загубите софтуера и данните на системата си, ако не извършите процедурата по правилния начин.

Да приемем, че в компютъра ви има един твърд диск, който изцяло се заема от MS-DOS. Следователно, вашият диск има един единствен дял на MS-DOS, който обикновено се нарича `C:`. Методът, който използваме, ще унищожи данните в този дял и затова трябва да създадете "системна дискета" на MS-DOS, от която може да се зарежда операционна

система. Тази дискета трябва да съдържа целия софтуер, който е необходим, за изпълнението на FDISK и за възстановяване на софтуера от архива, след като завършите преразпределянето на твърдия диск.

В много случаи за тази цел можете да използвате инсталационните дискети на MS-DOS. Ако все пак трябва да направите своя системна дискета, форматирайте една дискета с командата:

```
FORMAT /s A:
```

Копирайте на тази дискета всички необходими програми на MS-DOS (обикновено това е повечето от софтуера в директорията *\DOS* на вашия твърд диск), както и програмите *FORMAT.COM* и *FDlSK.EXE.* След като създадете дискетата, би трябвало да можете да заредите операционна система от нея и да изпълните командата:

```
FDISK C:
```

за да стартирате FDISK.

Използването на FDISK не би трябвало да ви затрудни, но все пак се консултирайте с документацията на MS-DOS за детайлите. Когато стартирате FDISK, използвайте командата от менюто му, с която се показват съществуващите дялове на твърдия диск и запишете изведената информация. Важно е да запишете оригиналните данни, за да можете при необходимост да възстановите началната конфигурация на системата си.

За да изтриете съществуващ дял, изберете от менюто на FDISK опцията **"Delete an MS-DOS partition or Logical DOS Drive"** (ИзтриЙдял на MS-DOS или логическо устройство на DOS). Задайте типа на дяла, който искате да изтриете (основен, "разширен" или логически) и номера му в отпечатаната таблица. Проверете дали не сте сбъркали и прочетете всички предупреждения. Бум!

За да създадете нов (по-малък) дял за MS-DOS, изберете командата на FDISK**"Create an MS-DOS Partition or Logical DOS Drive"** (Създай дял на MS-DOS или логическо устройство на DOS). Изберете типа на дяла (основен, "разширен" или логически) и размера му (числата са в мегабайти). FDISK би трябвало да създаде новия дял, след което можете да продължите.

След като завършите използването на FDISK, излезте от програмата и форматирайте новосъздадените дялове. Например, ако сте променили размера на първия дял на DOS на твърдия диск (c:) би трябвало да използвате командата:

```
FORMAT /s C:
```

Сега можете да възстановите оригиналния софтуер от архивното копие.

ГЛАВА 3

ИНСТАЛИРАНЕ И НАЧАЛНО КОНФИГУРИРАНЕ

Вече би трябвало да имате дистрибуция и да сте отделили дисково пространство за Linux. В тази глава ще направим общо описание на инсталирането и началното конфигуриране на Linux. Всяка дистрибуция има своите особености, но информацията, която ще получите в тази глава, би трябвало да се справите с която и да е инсталационна процедура. В Приложение А, *Източници на информация за Linux*, са изброени източници на информация за Linux, от които можете да получите помощ, ако възникне проблем, с който не можете да се справите.

Прилож.А

Различните дистрибуции на Linux записват файловете на различни места. Това затруднява създаването на общо описание на администрирането на Linux. Например, в Red Hat, SuSE и Debian използват някои еднакви файлове, но в една система те се намират в директорията */etc,* а в друга - в директорията */sbin.* Постепенно създателите на дистрибуции стандартизират местата на файловете в йерархията на директориите. Тези места са описани в документ, който се нарича Filesystem Hierarchy Standard (стандарт за йерархията на файловата система). В тази книга ще се опитаме да опишем все още съществуващите различия, като изброим местата на най-важните файлове във версиите на всяка основна дистрибуция, която сме проверили.

Инсталиране на софтуера за Linux

След като промените размера на съществуващите дялове на твърдия си диск, така че да освободите дисково пространство за Linux, можете да започнете да инсталирате софтуера. Следва кратко описание на тази процедура:

1. Зарежда се Linux от инсталационния носител.
2. Създават се дялове на Linux с програмата *fdisk* за Linux.
3. Създават се файлови системи за Linux и място за swap като се използват съответно програмите *mke2fs* и *mkswap*.
4. Инсталира се и се конфигурира софтуера.
5. Накрая, на твърдия диск се инсталира програмата за зареждане на Linux (LILO) или се създава дискета за зареждане на Linux.

Както вече споменахме, някои от тези стъпки могат да бъдат извършени автоматично от инсталационната програма. Това зависи от дистрибуцията, която използвате. За да получите конкретни инструкции за вашата дистрибуция, трябва да прочетете документацията й.

Зареждане на Linux

Първата стъпка е да заредите Linux от инсталационния носител. В повечето случаи, това е дискета, която съдържа малка Linux система, или компакт-диск, от който може да се зареди операционна система (bootable CD-ROM). След като се зареди Linux, автоматично се стартира програма, която ще ви ръководи, докато инсталирате софтуера. В някои дистрибуции при зареждане от дискета получавате покана за влизане в системата (login prompt). В такива случаи обикновено трябва да влезете като **root** или **install,** за да започнете инсталационната процедура.

Необходимите стъпки за зареждане на Linux от инсталационния носител са описани в документацията на вашата дистрибуция.

В повечето дистрибуции при зареждане на Linux от инсталационният носител можете да въведете конкретни параметри на хардуера. Тези параметри се използват, за да се улесни откриването на някои устройства в компютъра. Например, ако вашият SCSI контролер не бъде открит след зареждането на Linux, трябва да рестартирате машината и да укажете параметрите му (например входно/изходен адрес и rRQ). По същия начин при зареждане трябва да посочите параметрите на твърдия си диск, ако използвате компютър от сериите IBM PS/1, ThinkPad или ValuePoint, които не записват геометрията на диска в CMOS паметта си.

Често при зареждане от дискета или от компакт-диск автоматично се показва покана за въвеждане на команди - например в дистрибуцията на Red Hat. Ако вашата дистрибуция не показва автоматично тази покана, бихте могли да я получите като задържите натиснат клавиша Shift или Control при зареждане от инсталационния носител. Поканата би трябвало да изглежда по следния начин:

```
boot:
```

и вероятно още няколко съобщения. Това, което виждате на екрана, е поканата за въвеждане на команди (boot prompt) на LILO (съкращение от LInux LOader) - програма, която се използва за зареждане на Linux и задаване на параметри за разпознаване на хардуера при зареждането на операционната система. След като инсталирате Linux, бихте могли да инсталирате LILO на твърдия си диск и да я използвате, за да избирате коя операционна система да се стартира при зареждане на системата (например Linux или MS-DOS).

След като получите поканата на LILO, имате няколко възможности. Можете да натиснете клавиша Enter, за да заредите Linux без да задавате специални параметри (би трябвало да опитате първо този начин и ако инсталирането върви добре, значи всичко е наред). Можете просто да изчакате докато автоматично тръгне инсталационния процес - съвременните дистрибуции изчакват няколко секунди и ако потребителят не натисне клавиш през това време, те просто продължават нататък. Третата възможност е да въведете параметри, които да помогнат на инсталационния процес да идентифицира на хардуера на системата ви.

Ако не искате да въвеждате параметри на хардуера, когато получите подканата за въвеждане просто натиснете Enter. Наблюдавайте съобщенията, които се появяват, докато системата се зарежда. Например, ако имате SCSI контролер, би трябвало да видите списък с откритите SCSI-контролери. Ако системата изведе на екрана съобщение:

```
SCSI: 0 hosts
```

това означава, че SCSI контролерът ви не е открит и трябва да използвате процедурата за откриване на хардуер, която ще опишем след малко.

Повечето нови дистрибуции използват друг начин за избор на хардуер, при който инсталационният носител съдържа ядро с минимално количество вградени драйвери. Останалите драйвери се зареждат като *модули на ядрото* от втора дискета или от компакт-диск. В този случай най-вероятно ще ви се предостави меню, от което да изберете кои допълнителни модули да бъдат тествани. Дори този избор е автоматизиран до голяма степен - можете да поискате от инсталационната програма да се опита да разпознае SCSI контролерите във вашия компютър и да проверите дали вашият контролер е разпознат. Същото се отнася и за Ethernet картите и някои други устройства, които са необходими по време на инсталационния процес. Устройствата, които не са необходими по време на този процес (например звукови платки), обикновено не се разпознават на този етап. Вероятно инсталационната програма ще ви предостави възможност да ги конфигурирате по-късно.

Ако процедурите за автоматично разпознаване на хардуера не работят във вашата система (което обикновено се случва само ако имате много стар, съвсем нов или екзотичен хардуер), трябвало малко да помогнете на Linux, като директно въведете параметрите на хардуера си.

Конкретните параметри на хардуера могат да се въведат при зареждането на Linux, като се използва следният синтаксис:

linux *параметри*

Съществуват много такива параметри, някои от които са изброени подолу. Не е задължително да разбирате какво означава и за какво се използва всеки от тези параметри. Достатъчно е да можете да определите кои от тях са приложими за вашата система. Например, ако имате SCSI-контролер, базиран на чипа AHA 152x и знаете, че под MS-DOS трябва да го конфигурирате да работи с конкретен входно/изходен адрес и IRQ, можете тук да използвате съответната опция (aha152x=). Всъщност, повечето от опциите при зареждане на системата не са необходими за началното инсталиране.

Още един съвет: запишете и запомнете параметрите на хардуера, които използвате при инсталирането на Linux. След като завършите инсталирането, трябва да използвате същите параметри, за да може Linux да открива правилно хардуера при всяко зареждане. Ако инсталирате LILO на твърдия диск, можете да го конфигурирате така, че автоматично да зарежда Linux с необходимите опции, за да не се налага да ги въвеждате ръчно при всяко стартиране на Linux.

За ваше удобство ще опишем по-важните опции при начално зареждане на едно място:

no387

Забранява математическия копроцесор 80387. Използва се с някои копроцесори с хардуерни грешки, които не могат да работят в защитен режим.

no-hlt

Забранява използването на инструкцията HLT, с която процесора се установява в режим на работа с малка консумация на ток, когато системата не се използва. Някои от първите модели на чипа 486DX-100 имат проблеми при използването на тази инструкция.

root=устройство

Определя кое устройство ще се използва като основна файлова система при зареждането на Linux. Не трябва да се използва при инсталирането на системата. След инсталирането с тази опция можете да промените подразбиращото се място за основната файлова система.

ro Монтира основната файлова система в режим, при който е разрешено само четенето от нея. Използва се при поддръжката HaLinux.

lock

Записва параметрите при зареждане на системата за бъдеща употреба. По този начин не се налага да ги въвеждате всеки път, когато зареждате системата.

rw Монтира основната файлова система в режим, при който са разрешени четенето и писането в нея. Използва се при поддръжката на Linux.

debug

Докато системата работи, ядрото отпечатва на конзолата съобщения с подробна информация, която би могла да помогне за отстраняването на евентуални проблеми.

гатй18к=размер-в-кялобайти

Отделя указаното количество памет за виртуален диск (ramdisk). Често се използва в дискетите за зареждане на Linux, които зареждат в паметта цяла файлова система. Не би трябвало да използвате тази опция по време на инсталирането на Linux, но по-късно можете да я използвате, за да експериментирате с виртуални дискове.

тет=размер

Системният BIOS на повечето персонални компютри открива до 64 MB от инсталираната RAM памет. Linux използва тази информация, за да установи количеството налична памет. Ако имате повече от 64 MB памет и използвате старо ядро, трябва да зададете този параметър, за да може ядрото да използва системна памет след 64-я мегабайт. Размерът е число, към което могат да се добавят буквите k и м; например **mem=96M** означава, че в компютъра има инсталирани 96 MB RAM памет. Забележка: ако излъжете системата, че разполага с повече памет, отколкото има в действителност, ще се случат Лоши Неща.

Ъ.&=цилиндри, глави, сектори

Задава геометрията на твърдия диск с IDE или ST-506 интерфейс (но не и на SCSI-диск). Тази опция е задължителна за компютри от сериите IBM PS/1, ValuePoint и ThinkPad. Например, ако твърдият диск има 683 цилиндъра, 16глави и 32 сектора на пътека, задайте:

```
ramdisk hd=683,16,32
```

Тази опция има варианти - използвайте **hda=, hdb=, hdc=** или **hdd=,** за да опишете геометрията на конкретен IDE диск. Забележка: тази опция може да ви е необходима, ако използвате големи IDE дискове (с повече от 1024 цилиндъра). Ако Linux има проблеми с разпознаването на геометрията на вашия диск (ще разберете това, когато се опитате да създадете дялове за Linux), опитайте тази опция.

max_scsi_luns=yjrGtfo

Ако зададете 1, системата няма да се опита да открие SCSI устройства, които имат различен от 0 Logical Unit Number (LUN - логически номер на устройството). Този параметър е необходим при използването на някои SCSI устройства с лош дизайн, които блокират при използване на различен от 0 LUN. Забележка: LUN няма нищо общо със SCSI device ID (идентификатор на SCSI устройството). LUN се използва, за да се адресират различни логически устройства в конкретно SCSI устройство (например в дисково устройство).

r

aha15 2 x= ***i obase,irg,scsi-id,reconnect, parity***

Задава параметрите на SCSI контролерите Adaptec AHA151x, AHA152x, AIC6260, AIC6230 и SB16. ***iobase*** е шестнадесетичен запис на номера на базовия входно/изходен порт (например 0x340). Всички аргументи с изключение на ***iobase*** не са задължителни.

aha1542=iobase

iobase е шестнадесетично число, което задава базовия входно/изходен порт на SCSI контролерите Adaptec AHA154x.

aLc1xxx=extended, no-reset

Задава параметри за SCSI контролерите Adaptec AHA274x, AHA284x и AIC7xxx. Ако ***extended*** има различна от нула стойност, това означава, че трябва да се използва специална (разширена) транслация за твърдите дискове с голям размер. Ако стойността на ***no-reset*** е различна от нула, драйверът няма да инициализира SCSI шината при зареждането на системата.

buslogic=ioi>ase

iobase е шестнадесетично число, което задава базовия входно/изходен порт на SCSI контролерите Buslogic.

tmc8xx=mem-base, irq

Задава базовия адрес на паметта (шестнадесетично) и номера на IRQ, които се използват от SCSI контролерите Future Domain TMC-8xx и TMC-950.

pas16=iobase,irq

Задава базовия входно/изходен адрес (шестнадесетично) и номера на IRQ, които се използват от SCSI контролерите Pro Audio Spectrum.

st0x=mem-base, irq

Задава базовия адрес на паметта (шестнадесетично) и номера на IRQ, които се използват от SCSI контролерите Seagate ST-Ox.

t128=mem-base, irq

Задава базовия адрес на паметта (шестнадесетично) и номера на IRQ, които се използват от SCSI контролерите Trantor T128.

aztcd=iobase

Задава базовия входно/изходен адрес (шестнадесетично) на CD-ROM контролерите Aztech.

cdu3\a=iobase, irq,pas

Задава базовия входно/изходен адрес (шестнадесетично) и номера на IRQ за CD-ROM контролерите на Sony CDU-31A и CDU-33A. Тези опции се използват и за някои звукови карти Pro Audio Spectrum. Параметрите ***irq*** и ***pas*** не са задължителни. Ако ***irq*** е 0, не се използват прекъсвания (какъвто е случаят с някои звукови карти). Единствената валидна стойност за опцията ***pas*** е PAS, което означава, че се използва контролер Pro Audio Spectrum.

soncd535=ioAase,*irq*

Задава базовия входно/изходен адрес (шестнадесетично) и номера на IRQ (опционално) на CD-ROM контролерите Sony CDU-535.

gscd=iobase

Задава базовия входно/изходен адрес (шестнадесетично) на CD-ROM контролерите GoldStar.

mcd=iobase,irq

Задава базовия входно/изходен адрес (шестнадесетично) и (опционално) номера на IRQ на CD-ROM контролерите, отговарящи на стандарта Mitsumi.

optcd=io&ase

Задава базовия входно/изходен адрес (шестнадесетично) на CD-ROM контролерите Optics Storage.

cm206 = iobase, irq

Задава базовия входно/изходен адрес (шестнадесетично) и номер на IRQ на CD-ROM контролерите Philips CM206.

sjcd=iobase, irg, dma

Задава базовия входно/изходен адрес (шестнадесетично), номер на IRQ и DMA канал на CD-ROM контролерите Sanyo. Параметрите ***irgVL dma*** не са задължителни.

sbpcd=iobase, type

Задава базовия входно/изходен адрес (шестнадесетично) за CD-ROM контролери, съвместими със SoundBlaster Pro. Параметърът ***type*** трябва да бъде SoundBlaster, LaserMate или SPEA, в зависимост от типа на платката, която имате. Забележка: Този параметър на ядрото се отнася само за CD-ROM контролера, а не за звуковия хардуер на платката.

etber=irg,iobase,napaMeTpn...

Задава номера на IRQ и базовия входно/изходен адрес за Ethernet карти. Ако имате проблеми при откриването на вашия Ethernet контролер, но искате да го използвате за инсталиране на Linux (например като използвате протокол FTP или NFS), прочетете документа Linux Ethernet HOWTO, в който детайлно са описани останалите параметри за Ethernet карти. Тези параметри са твърде много, за да бъдат описани тук.

floppy= ***thinkpad***

Съобщава на дискетния драйвер, че имате компютър ThinkPad. Това е необходимо, за да имате достъп до дискетното устройство на тези платформи.

floppy=0, ***thinkpad***

Съобщава на дискетния драйвер, че нямате компютър ThinkPad, в случай, че драйверът погрешно разпознае компютъра ви като ThinkPad.

bmouse=irg

Задава номера на lRQ на контролера на busmouse.

msmouse=irg'

Задава номера на IRQ за Microsoft busmouse.

Съществуват още много други опции. Тук изброихме само тези, които обикновено са необходими за нормалната работа на системата. (Например, пропуснати са много опции, които се използват за настройка на

*

Busmouse е мишка, която се свързва към системната шина, вместо към серийния или PS/2 порта на компютъра.

драйверите за звукови карти. Ако работата на звуковата карта е въпрос на живот и смърт, най-добре прочетете подходящия HOWTO документ).

Ако имате въпроси за опциите при зареждане на системата, прочетете документите Linux Bootprompt HOWTO, Linux SCSI HOWTO и Linux CD-ROM HOWTO. Тези три документа описват детайлно аргументите на LILO при зареждане на системата. Можете да ги откриете на всеки FTP-сайт с информация за Linux (както и на повечето компакт-дискове със софтуер за Linux). "

[69]SCSI HOWTO
P1]CD-R0M H0WTO

Дискови устройства и техните дялове под Linux

При много дистрибуции се налага "ръчно" да създадете дяловете за Linux, като използвате *fdisk*. Други дистрибуции могат автоматично да създадат необходимите дялове. И в двата случая трябва да знаете някои особености за имената на дисковите устройства и техните дялове под Linux. (Тази информация се отнася само за платформи с процесори Intel или Alpha; други системи като PowerPC, SPARC и m68k нямат логически и "разширени" дялове.)

Под Linux дисковите устройства и дяловете в тях получават имена, които са различни от имената им в други операционни системи. Под MS-DOS дискетните устройства се наричат `A:` и `B:`, а дяловете на твърдите дискове се наричат `C:, D:` и т.н. Под Linux се използва съвсем различна схема за имената на устройствата.

В директорията */dev* се намират така наречените *драйвери на устройства,* които се използват за комуникация с хардуерните устройства във вашата система (твърди дискове, дискетни устройства и т.н.). Например, ако имате мишка, можете да комуникирате с нея като използвате драйвера */dev/mouse.* Дискетните устройства, твърдите дискове и отделните дялове в тях имат индивидуални драйвери. Засега не е необходимо да се задълбочавате в интерфейса на драйверите. Важното е само да знаете как се обозначават различните устройства, за да можете да ги използвате (виж раздела "Файлове на хардуерните устройства" в Глава 6).

Глава 6

Таблица 3-1 съдържа списък с имената на различните драйвери. Ако добавите число (0, 1 и т.н.) към името на драйвера, ще получите няколко имена. В таблицата има едно или две такива имена като примери.

Таблица 3-1: Имена на дяловете под Linux

Устройство	Име
Първо дискетно устройство (A:)	*/dev/fdO*
Второ дискетно устройство (B:)	*/dev/fdl*
Първи твърд диск (цялото устройство)	*/dev/hda*
Първи твърд диск, първи дял	*/dev/hdal*
Първи твърд диск, втори дял	*/dev/hda2*
Първи твърд диск, трети дял	*/dev/hda3*
Първи твърд диск, четвърти дял	*/dev/hda4*
Първи твърд диск, първи логически дял	*/dev/hda5*
Първи твърд диск, втори логически дял	*/dev/hda6*
Втори твърд диск (цялото устройство)	*/dev/hdb*
Втори твърд диск, първи дял	*/dev/hdbl*
Първи SCSI твърд диск (цялото устройство)	*/dev/sda*
Първи SCSI твърд диск, първи дял	*/dev/sdal*
Втори SCSI твърд диск (цялото устройство)	*/dev/sdb*
Втори SCSI твърд диск, първи дял	*/dev/sdbl*
Първо SCSI CD-ROM устройство	*/dev/scdO*
Второ SCSI CD-ROM устройство	*/dev/scdl*
Първо SCSI устройство от общ тип (скенер, CDR-устройство и т.н.)- В по-новите системи се използват цифри вместо букви (тоест */dev/sgO* вместо */dev/sga).*	*/dev/sga*
Второ SCSI устройство от общ тип	*/dev/sgb*

Няколко пояснения относно тази таблица: */dev/fdO* отговаря на първото дискетно устройство (A: под MS-DOS), а */dev/fdl* отговаря на второто дискетно устройство (в:). Освен това, имената на SCSI дисковете са различни от имената на останалите типове дискове. Достъпът до дисковете с интерфейс IDE, MFM и RLL е през устройствата */dev/hda, /dev/hdb* и т.н. Отделните дялове на диска */dev/hda* са */dev/hdal, /dev/hda2* и т.н.

Достъпът до твърдите дискове с интерфейс SCSI е през */dev/sda, /dev/sdb* и т.н., а дяловете в тях са */dev/sdal, /dev/sda2* и т.н.

Разбира се, повечето системи нямат четири основни дяла. Но имената */dev/hdal* до */devflida4* са запазени за тези дялове и не могат да се използват за имена на логически дялове.

Един пример. Да приемем, че имате един твърд диск с IDE интерфейс, който има три дяла. Първите два са отделени за MS-DOS, а третият е "разширен" дял и съдържа два логически дяла, които се използват от Linux. Файловете, които обозначават тези дялове, ще имат следните имена:

Устройство	**Име**
Първи дял на MS-DOS (C:)	*/dev/hdal*
Втори дял на MS-DOS (D:)	*/dev/hda2*
"Разширен" дял	*/dev/hda3*
Първи (логически) дял на Linux	*/dev/hda5*
Втори (логически) дял на Linux	*/dev/hda6*

Забележете, че устройството */dev/hda4* е прескочено; то съответства на четвъртия основен дял на диска, който в конкретния случай не съществува. Имената на логическите дялове се образуват последователно, като винаги се започва от */dev/hda5.*

Създаване на дялове за Linux

Вече имате достатъчно знания, за да създадете дялове за Linux с командата *fdisk*. В общия случай трябва да създадете най-малко един дял за софтуера на Linux и още един дял за swap.

Тук ще опишем използването на *fdisk* в текстов режим, което можете да използвате във всяка дистрибуция. Много от съвременните дистрибуции предлагат по-удобен и приятен интерфейс към *fdisk*. Макар че този интерфейс обикновено не е толкова гъвкав колкото обикновения, с него се работи по-лесно. Каквато и програма да използвате, прочетете този раздел, за да научите основните понятия и принципи за създаване на дялове в Linux. В крайна сметка, всички програми от типа на *fdisk* вършат една и съща работа, но някои от тях имат по-приятен интерфейс от останалите. Информацията в този раздел ще ви бъде полезна и когато използвате аналогична на *fdisk* програма с графичен интерфейс и искате да проверите или поправите резултата от нейните действия.

След като заредите от инсталационния носител, стартирайте *fdisk* като въведете командата:

```
fdisk drive
```

където ***drive*** е името (под Linux) на твърдия диск, в който смятате да създадете дял на Linux (виж Таблица 3-1). Например, ако искате да стартирате *fdisk* върху първия SCSI диск в компютъра ви, въведете командата:

 # fdisk /dev/sda

Ако не зададете дисковото устройство, по подразбиране се използва */dev/hda* (първият IDE диск).

Ако искате да създадете дялове на Linux на повече от един твърд диск, стартирайте *fdisk* по веднъж за всеки от тях:

 # fdisk /dev/hda

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

В този момент *fdisk* очаква команда. Можете да въведете m, за да получите списък с възможните команди:

```
Command (m for help): m
Command action
   a   toggle a bootable flag
   d   delete a partition
   1   list known partition types
   m   print this menu
   n   add a new partition
   p   print the partition table
   q   quit without saving changes
   t   change a partition's system id
   u   change display/entry units
   v   verify the partition table
   w   write table to disk and exit
   x   extra functionality (experts only)
Command (m for help):
```

Командата n се използва за създаване на нов дял. Повечето от останалите команди не са ви необходими в момента. За да излезете от *fdisk* без да записвате промените, които сте направили, използвайте командата q. Когато искате да запишете направените промени, излезте от *fdisk* като използвате командата w. Ще повторим - когато излизате от *fdisk* с командата q, не записвате промените, които правите. Това ви позволява да експериментирате колкото желаете с *fdisk,* без да рискувате данните на твърдия си диск. Потенциална опасност за данните ви съществува, само когато сбъркате и запишете промените си с командата w.

Първото нещо, което трябва да направите, е да установите какви дялове има на твърдия ви диск и да запишете тази информация на хартия. За да отпечатате на екрана таблицата с дяловете на диска, използвайте командата p. Добре е след всяка промяна, която правите, да записвате в бе-

лежника си информацията, която се извежда от командата p. Ако поради някаква причина се повреди таблицата с дяловете на диска, няма да имате достъп до данните в него, дори ако самите данни не са унищожени. Но като използвате бележките си, в повечето случаи ще можете да възстановите таблицата с дяловете на диска и да си върнете данните. За целта трябва да пуснете *fdisk* и отново да създадете дяловете с параметрите, които преди това сте си записали. Не забравяйте, че трябва да запишете на диска възстановената таблица с дялове.

Ето как се отпечатва една примерна таблица с дялове:

```
Command (m for help): p

Disk /dev/hda: 16 heads, 38 sectors, 683 cylinders
Units = cylinders of 608 * 512 bytes
   Device Boot   Begin    Start      End   Blocks   Id  System
/dev/hda1    *       1        1      203    61693    6  DOS 16-bit >=32M

Command (m for help):
```

В този пример в твърдия диск */dev/hda1* имаме един дял на MS-DOS, който има 61693 блока (около 60 MB). Този дял започва от цилиндъра с номер 1 и завършва на цилиндъра с номер 203. На диска имаме общо 683 цилиндъра, следователно остават свободни 480 цилиндъра, в които можем да се създадем дялове за Linux.

За да създадете нов дял, използвайте командата n. В този пример ще създадем два основни дяла за Linux *(/dev/hda2* и */dev/hda3):*

```
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
   p
```

fdisk пита какъв тип дял да създаде - "разширен" или основен. В този случай ще създаваме само основни дялове, затова избираме p:

```
Partition number (1-4):
```

Сега *fdisk* пита кой пореден номер ще бъде дяла, който създаваме. Понеже дялът на MS-DOS е с пореден номер 1, първият дял за Linux ще бъде с пореден номер 2:

```
Partition number (1-4): 2
First cylinder (204-683):
```

Сега трябва да въведем номера на цилиндъра, от който започва дялът. Понеже цилиндрите от 204-и до 683-и не са заети, ще използваме пър-

*

В Linux един блок е 1024 байта.

вия свободен от тях (с номер 204). Няма причина да оставяме празно място между дяловете:

```
First cylinder (204-683): 204
Last cylinder or +size or +sizeM or +sizeK (204-683):
```

fdisk пита колко голям ще бъде дялът, който искаме да създадем. Можем да въведем номера на цилиндъра, на който ще завърши дяла, или размера му в байтове, килобайтове или мегабайтове. Искаме нашият дял да бъде с размер 80 MB, затова въвеждаме +80M. Когато задаваме размера на дяла по този начин, *fdisk* ще закръгли действителния размер към най-близкия брой цилиндри:

```
Last cylinder or +size or +sizeM or +sizeK (204-683): +80M

Warning: Linux cannot currently use 33090 sectors of this partition
```

Ако видите подобно предупреждение, можете да го игнорирате. Такова съобщение се отпечатва от най-старите версии на *fdisk,* създадени във времената, в които дяловете на Linux са били с размер до 64 MB.

Сега можем да създадем и втория дял за Linux. Понеже това е само демонстрация, той ще бъде с размер 10 MB:

```
Command (m for help): n
Command action
   e   extended
   P   primary partition (1-4)
P
Partition number (1-4): 3
First cylinder (474-683): 474
Last cylinder or +size or +sizeM or +sizeK (474-683): +10M
```

Накрая ще изведем на екрана таблицата с дяловете на диска. Отново повтаряме, запишете в бележника си тази информация - особено размера на новите дялове. Този размер ще ви бъде необходим, когато създавате файловите системи. Освен това трябва да проверите дали вашите дялове не се застъпват:

```
Command (m for help): p

Disk /dev/hda: 16 heads, 38 sectors, 683 cylinders
Units = cylinders of 608 * 512 bytes
   Device Boot   Begin    Start     End   Blocks   Id  System
/dev/hda1    *       1        1     203    61693    6  DOS 16-bit >=32M
/dev/hda2          204      204     473    82080   83  Linux native
/dev/hda3          474      474     507    10336   83  Linux native
```

Както виждате, сега */dev/hda2* е дял, в който има 82080 блока (което е приблизително 80 MB), *a/dev/hda3* има 10336 блока (около 10 MB).

Забележка: Повечето дистрибуции на Linux изискват да използвате дял от типа "Linux swap" (Id 82) като дял за swap. Можете да използвате командата 1, за да отпечатате списък с познатите на *fdisk* кодове на дялове

и след това да използвате командата t, за да смените типа на избран от вас дял от "Linux native" на "Linux swap".

По този начин инсталационният софтуер ще може автоматично да открие вашия дял за swap, като използва типа му. Ако инсталационният софтуер не успее да разпознае съществуващия дял за swap, стартирайте отново *fdisk* и използвайте командата t върху въпросния дял.

В нашия пример останалите цилиндри на диска (с номера от 508 до 683) остават неизползвани. Можете да оставите това пространство неизползвано, ако искате по-късно да можете да създавате допълнителни дялове.

Накрая, ще използваме командата w, за да запишем направените промени на диска и да излезем от програмата *fdisk:*

```
Command (m for help): w
#
```

Не забравяйте, че промените, които сте правили във *fdisk,* нямат ефект, докато не въведете командата w. Така че, можете да опитвате различни конфигурации и да ги записвате, когато сте готови. Ако искате да прекъснете редактирането и да не записвате промените на диска, по всяко време можете да използвате командата q. Не забравяйте, че с *fdisk* за Linux не трябва да променяте дялове на други операционни системи.

Възможно е да не успеете да заредите Linux от дял, който използва цилиндри с номера по-големи от 1023. Затова трябва да се опитате да създадете основния дял на Linux в тази част от диска, която се намира преди 1024-я цилиндър. Това почти винаги е възможно (например като създадете малък дял преди 1024-я цилиндър). Ако поради някаква причина не можете или не искате да направите това, можете просто да заредите Linux от дискета.

Някои дистрибуции на Linux изискват рестартиране на компютъра, след като се извършат промени на диска с *fdisk,* за да могат направените промени влизат в сила преди да се инсталира софтуерът. Новите версии на *fdisk* автоматично опресняват информацията в ядрото за дяловете на диска и рестартирането на компютъра не е наложително. Ако искате да сте напълно сигурни, след като направите промени на диска с *fdisk,* рестартирайте компютъра преди да продължите с инсталирането на софтуера.

Създаване на място за swap

Глава 6

Ако смятате да използвате дял за swap за реализация на виртуална памет, сега можете да го подготвите. В раздела "Управление на виртуалната памет" в Глава 6 се описва как се създава swap-файл, в случай, че не искате да използвате отделен дял като място за swap.

Много дистрибуции изискват от вас да създадете и активирате място за swap, преди да инсталирате софтуера. Ако имате малко количество физическа памет, инсталационната процедура може да не приключи успешно. В такъв случай е необходимо да създадете и да активирате известно количество място за swap.

Командата *mkswap* подготвя (форматира) swap-дяла за работа и има следната форма:

```
mkswap -c partition size
```

където ***partition*** е името на дяла за swap, а ***size*** е размера на този дял в блокове.[г] Например, ако вашият дял за swap е */dev/hdaì* и се състои от 10336 блока, използвайте командата:

```
# mkswap -c /dev/hda3 10336
```

Опцията *-c* съобщава на *mkswap,* че трябва да провери дали дялът на диска съдържа лоши (неизползваеми) блокове, когато го форматира като място за swap. Лошите блокове са места на магнитния носител, които не могат да съхраняват коректно записаните в тях данни. При съвременните твърди дискове лошите блокове се срещат рядко, но ако вашият диск има този проблем, а вие не знаете за него, можете да имате безкрайни проблеми. Препоръчваме ви винаги да използвате опцията *-c,* за да проверите дали във вашия дял за swap има лоши блокове, *mkswap* автоматично ще забрани използването на такива блокове.

Ако използвате няколко дяла за swap, трябва да изпълните командата *mkswap* за всеки от тях.

След като форматирате мястото за swap, трябва да го активирате, за да може системата да го използва. Обикновено мястото за swap се активира автоматично при зареждането на системата. Но сега трябва да го активирате "ръчно", защото Linux все още не е инсталиран.

* Някои дистрибуции на Linux правят това автоматично или след команда от инсталационното меню.

[f] Това е размерът, който показва *fdisk* при въвеждането на командата p. Напомняме ви, че в Linux един блок е 1024 байта.

Командата за активиране на виртуалната памет е *swapon* и се използва по следния начин:

```
swapon partition
```

В нашия пример, след като изпълните командата *mkswap,* можете да активирате виртуалната памет със следната команда:

```
# swapon /dev/hda3
```

Създаване на файлови системи

За да можете да използвате някой от дяловете на Linux за съхранение на файлове, трябва да създадете файлова система в него. Създаването на файлова система е аналогично на форматирането на дял под MS-DOS или друга операционна система. Файловите системи бяха описани накратко в раздела "Изисквания за дяловете на Linux" в Глава 2, *Подготовка за инсталиране на Linux.*

Глава 2

Съществуват няколко типа файлови системи, които можете да използвате за Linux. Всеки тип има свой собствен формат и характеристики (максимална дължина на името на файл, максимален размер и т.н.). Linux поддържа и няколко типа файлови системи на други производители, например файловата система на MS-DOS.

Най-често използваният тип файлова система е *Second Extended Filesystem* (втора разширена файлова система) или *ext2fs. extìfs* е една от най-ефективните и гъвкави файлови системи. Тя позволява да използвате имена на файлове с дължина до 256 символа, а размерът ѝ може да достигне 4 терабайта. Различните файлови системи, които могат да бъдат използвани под Linux, са разгледани в раздела "Типове файлови системи" в Глава 6. Все пак, в началото ви препоръчваме да използвате файловата система *extlfs.*

Глава е

За да създадете файлова система *ext2fs,* използвайте следната команда:

```
mke2fs -c partition size
```

където ***partition*** е името на дяла, а ***size*** е размера на дяла в блокове. Например, за да създадете файлова система, която се състои от 82080 блока и се намира в */dev/hda2,* трябва да използвате командата:

```
I mke2fs -c /dev/hda2 82080
```

Ако използвате няколко дяла за Linux, трябва да изпълните командата *mkeìfs* със съответните параметри за всеки дял.

Ако на този етап възникнат някакви проблеми, вижте раздела "Какво да правим, ако възникнат проблеми" по-долу в тази глава.

Инсталиране на софтуера

Вашият компютър вече е готов за инсталирането на софтуера. Всяка дистрибуция използва различен механизъм за извършване на инсталирането. Много дистрибуции имат цялостна програма, която ви ръководи по време на инсталирането и може да извърши всички необходими стъпки. В други дистрибуции трябва сами да *монтирате* файловите системи в определена поддиректория (например */mnt)* и след това ръчно да копирате софтуера в тях. Някои от дистрибуциите на компакт-диск предоставят възможност да инсталирате само част от софтуера на твърдия си диск; останалият софтуер може да се използва директно от компакт-диска. Такова решение често се нарича live filesystem (буквално "жива" файлова система). То е удобно за тези, които искат да опитат Linux, преди да решат да инсталират всичко на твърдия си диск.

Някои дистрибуции предлагат няколко различни начина за инсталиране на софтуера. Например, бихте могли да инсталирате софтуера директно от MS-DOS дял на твърдия си диск, вместо да използвате дискети. Понякога можете да инсталирате Linux през TCPЯP мрежа с един от протоколите FTP и NFS. За повече подробности прочетете документацията на дистрибуцията, която използвате.

Например, дистрибуцията Slackware изисква от вас да извършите следното:

1. Трябва да създадете дялове за Linux с *fdisk*.

2. При желание можете да създадете дял за swap като използвате командите *mkswap* и *swapon* (препоръчваме ви да използвате виртуална памет, ако имате по-малко от 16 MB RAM памет).

3. Стартирайте програмата *setup,* за да инсталирате софтуера, *setup* ще ви преведе през системата от менюта, чрез които можете да зададете желаните от вас параметри и ще извърши останалата част от инсталационната процедура автоматично.

Точната процедура, по която се инсталира софтуерът, е специфична за всяка дистрибуция.

Можете да се смаете от количеството софтуер, който се предлага във всяка дистрибуция. Модерните дистрибуции на Linux често съдържат повече от хиляда пакета със софтуер и се състоят от няколко компакт-диска. Съществуват три основни метода, по които можете да изберете кои софтуерни пакети да инсталирате:

По целта, за която искате да използвате системата си.

Това е най-лесният начин за инсталиране и е подходящ за начинаещи. Не е необходимо да избирате всеки конкретен пакет - доста-

точно е да зададете целта, за която ще използвате компютъра си - като работна станция, машина за разработка на софтуер или мрежов маршрутизатор, и инсталационната програма автоматично ще избере необходимите пакети. При всички случаи след това ще можете да разгледате избраните пакети и ръчно да добавите или изключите някои от тях или да се върнете към инсталационната програма.

Избор на комплекти пакети

Ако използвате този вариант, ще можете да избирате групи пакети, които са предназначени за конкретни цели. Например "Работа в мрежа", "Разработка на софтуер" или "Графика". Възможно е да изберете индивидуални пакети от всяка група. Този вариант е малко по-труден от предишния, защото ще се наложи да избирате дали определена група пакети ви е необходима или не. Все пак можете да прескочите цели групи пакети, когато сте сигурни, че не се нуждаете от функционалността, която те ви предлагат.

Избор на индивидуални пакети

Трябва да използвате този метод, само ако знаете точно кои пакети искате да инсталирате. В противен случай, няма да видите гората заради дърветата.

Изборът на някой от изброените начини не изключва възможността да използвате и останалите. Повечето дистрибуции предлагат поне два от изброените механизми за избор на софтуер.

Вероятно все още ви е трудно да решите кои пакети да изберете. За да ви улеснят, добрите дистрибуции показват на екрана кратко описание на всеки пакет. Ако все още не сте сигурни, нашият съвет е следният: ако не сте сигурни дали някой конкретен пакет ви трябва или не, просто не го инсталирайте. Винаги можете да добавите нови софтуерни пакети към системата си.

Модерните дистрибуции имат една много полезна възможност - така наречената *проследяване па зависимостите.* Някои пакети работят само, ако са инсталирани определени други пакети (например, някои от програмите за показване на картини се нуждаят от специални графични библиотеки, с които се четат различните типове графични файлове). Като използва проследяване на зависимостите, инсталационната програма може да ви съобщи за тези зависимости и ви позволява автоматично да изберете пакета, който искате, заедно с всички пакети, от които той зависи. Освен ако не сте абсолютно сигурни какво правите, би трябвало винаги да използвате тази възможност, за да нямате по-късно проблеми при работата си.

Инсталационните програми могат да ви помогнат да направите избора си и да избегнете грешки и по други начини. Например, инсталационна-

та програма може да откаже да започне инсталационния процес, ако премахнете някой пакет, който е жизнено важен за зареждането дори на най-простата система (например, основната структура на директориите). Или, програмата може да провери за взаимно изключващи се пакети - съществуват пакети, от които можете да изберете единия или другия, но не и двата пакета.

Някои дистрибуции (например SuSE) се разпространяват в комплект с голяма документация, която освен другата информация, съдържа списък с всички пакети заедно с кратките им описания. Би било добре поне бегло да прочетете тези описания, за да видите какво ви се предлага - в противен случай не се изненадвайте, ако след като изберете някой пакети, инсталирате 25-я пореден текстов редактор.

Как да изберем начина за зареждане на Linux

След като инсталирате софтуера, трябва да изберете начина за зареждане на новата си операционна система. Всяка дистрибуция предлага няколко решения. В повечето случаи, инсталационната процедура ви предлага да създаде дискета за зареждане на Linux, която ще съдържа ядрото на операционната система, конфигурирано да използва вашата току-що създадена основна файлова система. Бихте могли да използвате тази дискета за зареждане на Linux от дискетното устройство. След като ядрото се стартира от дискетата, зареждането ще продължи от твърдия диск. При някои дистрибуции дискета за зареждане на Linux е самата инсталационна дискета.

Много дистрибуции ви предоставят възможност да инсталирате LILO на твърдия си диск. LILO е програма, която се записва в първия сектор на твърдия диск (така наречения Master Boot Record или MBR - основен запис за начално зареждане). LILO може да зареди множество операционни системи (включително MS-DOS и Linux) и ви позволява да изберете коя от тях да се зареди при стартиране на компютъра.

За да се инсталира успешно, LILO трябва да знае доста неща за конфигурацията на твърдия ви диск - например, какви операционни системи съдържат различните дялове на диска, как да зареди всяка отделна операционни системи и т.н. Когато инсталират LILO, много дистрибуции се опитват да "отгатнат" подходящите параметри за вашия компютър. Макар и не особено често, автоматичната процедура за инсталиране на LILO може да не работи добре с вашата система и да повреди MBR сектора на твърдия диск (почти невъзможно е това да унищожи данните ви). По-конкретно, това може да се случи, ако използвате Boot Manager на OS/2 - в такъв случай *не трябва* да инсталирате LILO с автоматичната процедура; съществуват специални инструкции за това как да из-

ползвате LILO с Boot Manager (виж Глава 5, *Основи на системната администрация).*

В много случаи най-добре е да зареждате Linux от дискета, докато не получите възможност ръчно да конфигурирате LILO. Разбира се, ако имате пълно доверие на дистрибуцията си, можете да използвате автоматичната процедура за инсталиране на LILO.

Ще опишем детайлно конфигурирането и инсталирането на LILO на вашата система в раздела "Как да използваме LILO" в Глава 5.

Ако дотук всичко върви добре, приемете нашите поздравления. Току що инсталирахте Linux на компютъра си. Изпийте чаша чай или нещо друго - заслужили сте си го.

Ако са възникнали някакви проблеми, прочетете раздела "Какво да правим, ако възникнат проблеми" по-долу в тази глава. В него са описани най-често възникващите проблеми по време на инсталирането на Linux и начините за тяхното решаване.

Допълнителни инсталационни процедури

Част от дистрибуциите на Linux извършват някои допълнителни процедури, след като завърши инсталирането на софтуера. Това включва конфигуриране на различни софтуерни пакети като TCP/IP, системата X Window и т.н. Ако имате възможност да конфигурирате тези аспекти на системата си по време на инсталирането й, бихте могли да прочетете следващите глави в тази книга, за да научите как да конфигурирате този софтуер. Разбира се, ако искате, можете сега да прекъснете тези процедури и да ги отложите за момента, в който научите достатъчно за конфигурирането на софтуера.

Всичко зависи от вас. Ако всичко друго се провали, оставете се на течението и вижте какво ще се случи. Едва ли ще можете да направите нещо толкова лошо, че след това да не можете да поправите грешката си (да чукнем на дърво).

.

Процедури след инсталиране на Linux

След като завършите инсталирането на Linux, би трябвало да можете да рестартирате системата, да влезете в нея като **root** и да започнете да я изследвате. (Всяка дистрибуция използва различен метод за това; следвайте инструкциите на вашата дистрибуция.)

Все пак, преди да започнете, трябва да извършите няколко процедури, които ще ви спестят бъдещи неприятности. Някои от тези процедури ще ви се сторят тривиални, ако имате подходящ хардуер и добра дистрибуция на Linux; за други процедури може да се наложи да направите известно разучаване на системата и вероятно ще ги отложите за по-късно.

Регистриране на потребител

За да започнете да използвате системата си, трябва да регистрирате нов потребител, като създадете така наречения user account (описание на потребител) за себе си. Ако искате компютърът ви да се използва от няколко потребителя, бихте могли да ги регистрирате като създадете индивидуален account за всеки от тях. Но преди да продължите нататък, трябва да регистрирате поне един потребител.

Защо е нужно това? Всяка Linux система има няколко предварително регистрирани потребителя (тоест account-а), един от които е **root.** Този account е предназначен преди всичко за администриране на системата и когато го използвате, ще имате всякакви привилегии и ще имате пълен достъп до всеки файл в системата.

Използването на системата като **root** може да бъде опасно, особено за начинаещи потребители. Понеже не съществуват ограничения за действията на потребителя **root,** при използването на този account можете неволно да изтриете файлове, да повредите файловата си система и т.н. Би трябвало да използвате системата си като **root** само, когато трябва да извършите някаква административна задача, например настройка на конфигурационните файлове, инсталиране на нов софтуер и т.н. (виж раздела "Администриране на системата" в Глава 5).

Глава 5

За всекидневната си работа би трябвало да използвате Linux като обикновен потребител. Unix има вградена система за сигурност, която не позволява на потребителите да изтриват файлове на други потребители или да повредят важни ресурси като конфигурационните файлове на системата. Когато работите като обикновен потребител, се предпазвате от собствените си грешки. Това е особено важно за потребители, които нямат опит в системното администриране на Unix.

Много от дистрибуциите на Linux предоставят програми за регистриране на нови потребители. Тези програми обикновено се наричат *useradd* или ***adduser.*** Ако влезете в системата си като **root,** можете да стартирате една от тези програми, за да получите кратко описание на начина, по който можете да я използвате. Регистрирането на нов потребител би трябвало да е изключително лесно.

Повечето модерни дистрибуции предлагат инструмент за цялостна системна администрация, който може да се използва за много цели, една от които е регистрирането на нов потребител.

Други дистрибуции, като SuSE Linux и Caldera Open Linux, използват един и същи инструмент за инсталиране и за системна администрация - *yast* в SuSE или *lisa* в Caldera Open Linux.

Ако е необходимо, можете ръчно да регистрирате нов потребител. Обикновено за това е достатъчно да направите следното:

1. Редактирайте файла */etc/passwd,* за да добавите нов потребител.
2. Ако използвате скрити пароли (shadow passwords), трябва да редактирате файла */etc/shadow,* за да добавите в него атрибутите на новия потребител.
3. Създайте лична директория на потребителя.
4. Копирайте стандартните конфигурационни файлове (като *.bashrc)* в директорията на новия потребител. Понякога тези файлове се намират в директорията */etc/skel.*

Не бихме искали тук да описваме с големи подробности регистрирането на нов потребител - детайлите могат да бъдат намерени практически във всяка книга за администриране на Unix (виж Библиографията). Освен това, регистрирането на нов потребител е описано в раздела "Управление на потребителски account" в Глава 5. Ако имате късмет, във вашата дистрибуция ще можете да намерите програма, която да се погрижи за всички детайли вместо вас.

Не забравяйте да зададете (или смените) паролата на новия потребител. За тази цел трябва да използвате командата *passwd.* Например, за да смените паролата на потребителя **duck,** въведете следната команда:

 # passwd duck

Ако сте влезли като **root,** *passwd* ще ви попита каква да бъде новата парола на duck, без да пита за старата. По този начин, ако сте забравили паролата си, но все още можете да влезете в системата като **root,** ще можете да зададете нова парола за вашия account.

Как да използваме електронната документация

Linux разполага със система за електронна документация във формата на така наречените "справочни страници" (manual или man pages). На много места в тази книга ще ви препращаме към страниците на някои конкретни команди на Linux за по-подробна информация. Справочните

страници описват детайлно програмите и приложенията в системата. Затова е важно да се научите как да използвате тази електронна документация, когато ви се наложи.

За да получите справочната страница на някоя конкретна команда, използвайте командата *man*. Например, за да получите информация за командата *passwd,* въведете следното:

```
$ man passwd
```

Би трябвало да видите на екрана справочната страница за *passwd.*

Обикновено справочните страници се разпространяват като отделен пакет в повечето дистрибуции и за да могат да се използват, трябва да бъдат инсталирани. Горещо ви препоръчваме да ги инсталирате - това би ви спестило много главоболия.

Трябва да добавим, че някои страници могат да липсват или да са незавършени, в зависимост от това колко пълна е дистрибуцията и кога за последен път е обновявана електронната документация.

Освен това, в справочните страници за Linux са документирани програмният интерфейс и архитектурата на ядрото, системните библиотеки и форматът на конфигурационните файлове. За детайлно описание на използването на справочните страници виж раздел "Справочни страници" Глава 4 в глава 4^ *Основни команди и концепции на Unix.*

Освен традиционните справочни страници, съществуват и така наречените информационни страници (info pages). Те могат да се четат в текстовия редактор Emacs, с командата *info* или с някоя от множеството графични програми, предназначени за целта.

Много дистрибуции съдържат документация във формат HTML. Тази документация може да се чете с произволен web-браузър като Netscape Navigator или с текстовия редактор Emacs.

И накрая, съществува документация във вид на обикновени текстови файлове. Тази документация може да се чете с произволен текстов редактор или просто с командата *more.*

Ако не можете да намерите документацията за някоя конкретна програма, опитайте се да я пуснете с една от опциите ***-h*** или —. За повечето програми това означава команда за отпечатване на кратко описание на функциите им.

Редактиране на конфигурационния файл /etc/fstab

Възможно е да ви се наложи да редактирате конфигурационния файл */etc/fstab,* за да гарантирате достъпа до всички файлови системи след рестартиране на системата. В този файл са описани вашите файлови системи. Много дистрибуции автоматично генерират файла /etc/fstab по време на инсталирането си, така че вероятно няма да имате проблеми. Все пак, ако имате допълнителни файлови системи, които не са били използвани по време на инсталационния процес, трябва да ги опишете във файла */etc/fstab,* за да можете да ги използвате. В този файл трябва да бъдат описани и swap-дяловете в системата ви.

За да имате достъп до определена файлова система, тя трябва да се *"монтира".* Монтирането асоциира файловата система с конкретна директория. Например, основната файлова система се монтира като /, файловата система */usr* се монтира като директория */usr* и т.н. (Ако не сте създали отделна файлова система за */usi;* всичките файлове в тази йерархия ще бъдат записани в основната файлова система.)

Не бихме искали тук да ви затрупваме с технически детайли, но е важно да разбирате как можете да осигурите достъп до файловите си системи, преди да започнете да изследвате системата си. За по-подробно описание на монтирането на файлови системи прочетете раздела "Монтиране на файлови системи" в Глава 6 или някоя книга за системна администрация на Unix.

Основната файлова система се монтира автоматично като / при зареждането на Linux. Останалите файлови системи трябва да бъдат монтирани индивидуално. Обикновено това се прави с командата:

 # mount -av

в някой от автоматично изпълняваните системни файлове в */etc/red* или директорията, в която вашата дистрибуция съхранява конфигурационните си файлове. Параметрите означават, че командата *mount* трябва да се опита да монтира всички файлови системи, които са изброени във файла */etc/fstab.* Следователно, ако искате вашите файлови системи да се монтират автоматично при всяко стартиране на Linux, трябва да ги добавите в */etc/fstab* (разбира се, винаги можете ръчно да монтирате файловите системи с командата *mount,* но това е излишен труд).

Следва един примерен файл */etc/fstab.* В този пример основната файлова система се намира в */dev/hdal,* файловата система */home* се намира в */dev/hdb2,* а дяла за swap се намира в */dev/hdbl:*

```
# /etc/fstab
# device      directory   type    options
#
/dev/hda1                 ext2    defaults
/dev/hdb2     /home       ext2    defaults
/dev/hdb1     none        swap    sw
/proc         /proc       proc    defaults
```

Редовете, които започват със символа # се смятат за коментар и се игнорират. Забележете допълнителния ред за */proc*. */proc* е "виртуална файлова система", която се използва за получаване на информация за работещите процеси от команди като *ps*.

Както виждате, файлът */etc/fstab* се състои от последователност от редове. Първото поле на всеки ред съдържа името на дяла, например */dev/hda1*. Второто поле съдържа *мястото на монтиране* — директорията, в която се монтира файловата система. Третото поле съдържа типа на файловата система - например за файловите системи *ext2fs* трябва да зададете тип ext2; за дял за swap трябва да зададете тип swap и т.н. Четвъртото поле съдържа опциите при монтиране. Би трябвало да използвате defaults за обикновените файлови системи и sw за swap-дял.

Като използвате този пример, бихте могли да добавите дефиниция за всяка файлова система, която не е описана във файла */etc/fstab*.

Как да добавим редове към този файл? Най-лесният начин е чрез редактирането му. Трябва да влезете като **root** и да използвате някой текстов редактор, например *vi* или Emacs. Тук няма да описваме как се използват текстовите редактори. Можете да намерите описание на работата с *vi* и Emacs в началото на Глава 9, *Редактори, текстови инструменти, графика и печат*.

След като редактирате */etc/fstab* трябва да изпълните командата:

 # /bin/mount -a

или да рестартирате компютъра си, за да се монтират новите файлови системи.

Не се притеснявайте, ако възникнат някакви проблеми. Препоръчваме на начинаещите потребители да прочетат някоя книга, в която са описани основите на използването на Unix и системната му администрация. По-голямата част от следващите глави е написана като сме предполагали, че сте запознати с основите на Unix, затова не казвайте, че не сме ви предупредили.

Спиране на системата

Би трябвало никога да не рестартирате или изключвате Linux система като натискате бутона reset или просто като изключите захранването.

Подобно на повечето Unix системи, Linux използва така наречения "кеш" (cache) при запис върху диска. Това означава, че ако внезапно рестартирате компютъра, без да изпълните процедурата за "чисто" спиране (shut down) на Linux, бихте могли да повредите данните на дисковете си. Забележка: използването на клавишната комбинация Ctrl-Alt-Del не създава проблеми - ядрото разбира, че сте натиснали тези клавиши и автоматично стартира процедурата за чисто спиране на системата (виж Глава 5). Възможно е Linux да бъде конфигуриран по такъв начин, че комбинацията Ctrl-Alt-Del да бъде резервирана за системния администратор и да няма ефект, ако се натисне от обикновен потребител. Това е необходимо, за да се избегнат проблемите, които възникват, когато някой обикновен потребител спре мрежов сървър, от когото зависи работата на цял отдел. За да ограничите кръга от потребители, които могат да спират системата, трябва да създадете файл с име */etc/shutdown.allow,* в който да изброите имената им.

Най-лесният начин да спрете системата е да използвате командата *shutdown.* Като пример, за да започнете веднага процедурата за спиране на Linux и рестартиране на компютъра, влезте като root и изпълнете командата:

```
# shutdown -r now
```

По този начин системата ви ще бъде рестартирана чисто. В справочната страница за *shutdown* са описани и други параметри на тази команда. Вместо now ("сега" на английски), можете да укажете кога трябва да започне процесът по спиране на системата. При много дистрибуции можете да използвате командата *halt,* която изпълнява *shutdown now.* При някои дистрибуции съществува командата *poweroff,* с която можете да спрете Linux, след което компютърът автоматично ще се изключи. Тази команда работи само с машини, които имат необходимия хардуер (компютърът трябва да поддържа APM).

Какво да правим, ако възникнат проблеми

Почти всеки, който инсталира Linux за пръв път, има някакви проблеми. В повечето случаи причина за проблема е погрешното разбиране на определена концепция. Понякога обаче, проблемът може да е по-сериозен - например пропуск на дистрибуцията или дори програмна грешка в Linux.

В този раздел са описани някои от най-често срещаните проблеми при инсталирането на Linux и начините за решаването им. Освен това са

описани някои от проблемите, които възникват след успешно инсталиране на Linux.

Проблеми при зареждане от инсталационния носител

Понякога при първия си опит за зареждане от инсталационния носител можете да се сблъскате с проблеми. Забележка: тези проблеми *не са* свързани със зареждането на току-що инсталирана Linux система (виж раздела "Проблеми след инсталирането на Linux").

Грешка при четене от инсталационния носител при опит за зареждане

Най-честата причина за този тип проблеми е повредена дискета. Това може да се случи, ако дискетата има физически дефект - в такъв случай трябва да създадете инсталационната дискета отново като използвате чисто нов носител. Проблемът може да е причинен и от повредени данни - това може да се случи, ако сте изтеглили дистрибуцията от Интернет или е възникнал проблем при прехвърлянето на данните върху дискетата, затова трябва да проверите дали данните са коректно изтеглени и прехвърлени върху дискетата. В повечето случаи за решаване на проблема е достатъчно отново да създадете дискетата за зареждане на Linux. Проследете стъпките си и опитайте отново.

Ако сте получили дискетата си по пощата или сте я закупили по друг начин, можете да се свържете с дистрибутора и да поискате нова дискета - но само, след като се уверите, че проблемът действително е този.

Системата "увисва " по време или след зареждане.

След като операционната система се зареди от инсталационния носител, ядрото би трябвало да изведе съобщения, в които се описват откритите и конфигурирани устройства. Обикновено след това на екрана се извежда покана за влизане в системата (login prompt), която ви позволява да продължите инсталирането на Linux (съвременните дистрибуции вместо това автоматично стартират инсталираща програма от някакъв тип). Системата може да изглежда "увиснала" по време на някои от тези стъпки. Бъдете търпеливи - зареждането на софтуер от дискета е много бавно. В много случаи системата не е увиснала, просто зареждането продължава дълго време. Би трябвало да изминат поне няколко минути, през които дисковите устройства да не се използват и на екрана да не се извеждат съобщения, преди да приемете, че системата е "увиснала".

Правилната последователност на зареждане е следната:

1. След като изберете начин на зареждане от поканата на LILO, системата трябва да стартира ядрото от дискетата. Това може да отнеме няколко секунди. Ако индикаторът на дискетното устройство свети, вероятно всичко е наред.

2. След като се зареди, ядрото се опитва да открие SCSI устройствата в компютъра ви. Ако нямате инсталирани SCSI контролери, системата ще изглежда "увиснала" докато трае този процес (не повече от 15 секунди). Обикновено това става, след като на екрана се отпечата текстът:

   ```
   lp_init: lpl exists (0), using polling driver
   ```

3. След като завърши зареждането на ядрото, управлението се прехвърля на файловете за зареждане на дискетата. Накрая се извежда покана за влизане в системата или автоматично тръгва инсталиращата програма. Ако получите покана за влизане в Linux, като тази:

   ```
   Linux login:
   ```

   трябва да влезете в системата (обикновено като **root** или **install** - това зависи от дистрибуцията, която използвате). След като напишете потребителското име, са необходими 20 секунди или повече, за да се зареди инсталиращата програма. Индикаторът на дискетното устройство трябва да свети по време на зареждането. Не мислете, че системата е увиснала, докато трае този процес.

Всяко от изброените действия може да продължи известно време.

През това време системата не е спряла. Все пак, възможно е системата действително да "увисне" при зареждане - това може да се предизвика от няколко причини. На първо място, възможно е да нямате достатъчно физическа памет (виж следващата точка за информация как се забранява виртуалния диск, за да се освободи памет).

Много системи увисват заради хардуерна несъвместимост. Разделът "Хардуерни изисквания" в Глава 1 съдържа списък с хардуера, който се поддържа от Linux. Възможно е да имате проблеми, дори когато Linux поддържа вашия хардуер. Това се случва при някои несъвместими настройки на устройствата, които използвате (виж следващия раздел на тази глава, "Хардуерни проблеми").

Системата извежда съобщение за недостатъчна памет при зареждане или при опит за инсталиране на софтуера.

Проблемът е свързан с количеството физическа памет, с което разполагате. Ако компютърът ви има по-малко от 4 MB памет, можете да имате проблеми докато зареждате от инсталационния носител или инсталирате самия софтуер. Причината за проблема е, че много

дистрибуции използват "виртуален диск" (ramdisk), който представлява файлова система, заредена директно в паметта на компютъра. Виртуалният диск се използва за някои операции по време на инсталирането на Linux. Например, в него могат да се прехвърлят файловете от дискетата за зареждане на Linux, за което е необходим около 1,5 MB памет.

Решението на този проблем е при зареждане от инсталационния носител да се забрани използването на виртуален диск. Всяка дистрибуция има своя процедура за това. Например, при SLS можете да въведете floppy, когато получите поканата на LILO при зареждане от дискетата ***al.*** Повече подробности можете да намерите в документацията на дистрибуцията, която използвате.

Възможно е да не получите съобщение за недостиг на памет, а вместо това компютърът неочаквано да "увисне" или да не успее да завърши зареждането на операционната система. Ако системата ви увисне и нито едно от предложенията в предишната точка не помогне, опитайте се да забраните използването на виртуален диск.

Не забравяйте, че самото ядро изисква поне 4 MB памет, за да тръгне, а почти всички съвременни дистрибуции на Linux изискват 8 MB или повече RAM.

При зареждане системата извежда съобщение за грешка от типа "Permission denied" или "File not found".

Тази грешка е признак за повреден инсталационен носител. Ако сте закупили инсталационния носител (и сте сигурни, че не допускате грешка), не би трябвало да се получават такива грешки. Свържете се с дистрибутора, за да откриете какъв е проблемът и при необходимост да получите друг инсталационен носител. Ако сте изтеглили дистрибуцията си от Интернет, опитайте се отново да направите дискета за зареждане на Linux и проверете дали това няма да реши проблема ви.

При зареждане системата извежда следното съобщение за грешка "VFS: Unable to mount root".

Това съобщение означава, че ядрото не може да открие основната файлова система (която се намира на инсталационния носител). Грешката може да се получи, ако инсталационният носител е повреден или ако зареждате системата неправилно.

Например, много дистрибуции на компакт-диск изискват диска да е в компакт-дисковото устройство при зареждане на системата. Проверете дали самото компакт-дисково устройство е инсталирано коректно и дали е включено (за външни устройства). Възможно е при

зареждане си ядрото да не открива устройството ви. За повече информация виж следващия раздел "Хардуерни проблеми".

Ако сте сигурни, че зареждате системата си правилно, вероятно инсталационният ви носител действително е повреден. Това е рядко срещан проблем, затова ви съветваме да опитате някое от другите решения, преди да опитате с друг носител.

Хардуерни проблеми

Най-често срещаният проблем при инсталирането или използването на Linux е несъвместимост с хардуера. Възможно е да имате проблеми, дори когато Linux поддържа всяко устройство, което използвате. Обикновено причина за проблема е лоша конфигурация или хардуерен конфликт - като резултат устройствата не се откриват при зареждане или системата увисва.

Важно е тези хардуерни конфликти да се открият и изолират, ако смятате, че те са източника на проблема. В следващите раздели ще опишем някои от често срещаните хардуерни проблеми и начините за тяхното решаване.

Изолиране на хардуерни проблеми

Ако имате проблем, който според вас е свързан с хардуера в компютъра ви, първо трябва да се опитате да откриете откъде точно идва проблема. Това означава, че трябва да елиминирате всички възможности, (обикновено) като разглобите компютъра си, част по част, докато откриете именно коя част предизвиква проблема.

Това не е толкова страшно, колкото изглежда на пръв поглед. Най-общо казано, трябва да изключите компютъра си и да извадите всички устройства, които не са от жизнено значение за работата на машината и след това да установите кое е устройството, което причинява проблема. Това става като връщате едно по едно устройствата, които сте извадили. Обикновено трябва да извадите всичко с изключение на дискетното устройство, видео-контролера и клавиатурата. Дори безобидно изглеждащите устройства могат да ви съсипят нервите, ако ги сметнете за маловажни.

Нека вземем като пример компютър, който увисва при зареждане на Linux по време на разпознаването на мрежовия контролер. Бихте могли да предположите, че става въпрос за конфликт или несъвместимост между мрежовия контролер и друго устройство в машината ви. Найлесният начин да проверите дали това е така, е да извадите мрежовия контролер и да опитате отново да заредите системата. Ако след рес-

Глава 1

тартирането всичко върви добре, значи или мрежовият контролер не се поддържа от Linux (виж раздел "Хардуерни изисквания" в Глава 1), или се получава конфликт от входно/изходните адреси или номера на IRQ, които се използват от мрежовия контролер. Като допълнение ще кажем, че някои мрежови контролери с лош дизайн (основно имитации на NE2000) могат да накарат целия компютър да увисне по време на процедурата за автоматичното им разпознаване - ако смятате, че вашия мрежов контролер е такъв, препоръчваме ви да го извадите за времето на инсталирането и да го върнете в компютъра след това, или да зададете подходящи параметри на ядрото при зареждането на системата, така че да избегнете процедурата за автоматично разпознаване. Най-доброто решение е да изхвърлите злополучната мрежова платка и да закупите нова от производител, който разработва продуктите си по-внимателно.

Какво всъщност означава "хардуерен конфликт, предизвикан от входно/изходните адреси или номера на IRQ, които използва мрежовия контролер"? Повечето устройства в компютъра ви използват т.нар. *interrupt request line* или IRQ (линия за заявка за прекъсване), за да съобщят на системата, че трябва да извърши определени дейности по обслужването им. Можете да мислите за IRQ като за конец, който се дърпа от устройството, когато то се нуждае от внимание от страна на системата. Ако няколко устройства дърпат един и същи конец, ядрото не може да установи точно кое устройство се нуждае от обработка.

Следователно, трябва да се уверите, че всичките устройства използват различни номера на IRQ. В повечето случаи номера на IRQ, който се използва от дадено устройство, може да се зададе от превключватели (jumpers) на самата платка - за детайлите трябва да прочетете документацията на конкретното устройство. Някои устройства могат да работят и без да използват IRQ, но се препоръчва при възможност да бъдат конфигурирани така, че да използват IRQ (пример за това са SCSI контролерите Seagate ST01 и ST02).

В някои случаи ядрото, което се намира на инсталационния носител, е конфигурирано така, че да използва фиксирани номера на IRQ за определени устройства. Например, в някои дистрибуции на Linux се използва ядро, което е предварително конфигурирано така, че да използва IRQ 5 за SCSI контролера TMC-950, за CD-ROM контролера Mitsumi и за драйвера на busmouse. Ако искате да използвате две или повече от тези устройства, трябва първо да инсталирате Linux, като в машината ви има само едно от устройствата и след това да прекомпилирате ядрото, за да смените номера на IRQ, който се използва за другите устройства (виж раздел "Създаване на ново ядро" в Глава 7, *Актуализиране на софтуера и ядрото).*

Глава 7

Хардуерен проблем може да възникне и при неправилна настройка на каналите за *директен достъп до паметта* (DMA или direct memory access), входно/изходните адреси (I/O address) или адреса на паметта на устройствата (shared memory address). Всичките тези термини описват механизмите, по които системата комуникира с хардуерните устройства. Например, при някои Ethernet карти част от адресното пространство на процесора се използва за комуникация с паметта на картата. Ако друго устройство използва същото адресно пространство, поведението на системата е непредсказуемо. Би трябвало да можете да смените DMA каналите, входно/изходните адреси и адреса на паметта на устройствата си с настройка на превключвателите им (за съжаление, някои устройства не позволяват промяна на тези настройки).

Документацията на хардуера, който използвате, би трябвало да описва номера на IRQ, DMA каналите, входно/изходните адреси и адреса на паметта, които използва устройството, заедно с начина на конфигуриране на тези параметри. Още веднъж ще кажем, че най-лесния начин да заобиколите този тип проблеми е временно да извадите устройствата, които създават проблеми, докато намерите време да установите източника на проблема.

Таблица 3-2 съдържа списък с номерата на IRQ и DMA каналите, използвани от някои от "стандартните" устройства, които могат да бъдат открити в повечето компютри. Почти всички компютри използват някои от тези устройства, затова трябва да избягвате да настройвате номерата на IRQ или DMA на други устройства с тези стойности.

Таблица 3-2: Често срещани настройки на някои хардуерни устройства

Устройство	В/И адрес	IRQ	DMA
ttySO (COMI)	3F8	4	
ttySl (COM2)	2F8	3	
ttyS2 (COM3)	3E8	4	
ttyS3 (COM4)	2E8	3	
IpO (LPT1)	378- 37F	7	
Ipl (LPT2)	278 - 27F	5	
fdO.fdl (дискетни устройства 1 и 2)	3F0 - 3F7	6	2
fd2,fd3 (дискетни устройства 3 и 4)	370 - 377	10	3

Проблеми при разпознаването на твърдия диск или контролера му

Когато се зарежда, Linux отпечатва множество съобщения на екрана на компютъра ви. Например:

```
Console: colour EGA+ 80x25, 8 virtual consoles
Serial driver version 3.96 with no serial options enabled
ttyOO at 0x3f8 (irq = 4) is a 16450
tty03 at 0x2e8 (irq = 3) is a 16550A
lp_init: lpl exists (0), using polling driver
```

В този пример ядрото разпознава различни хардуерни устройства в компютъра ви.' Би трябвало едно от всичките съобщения да е следното:

```
Partition check:
```

последвано от списък с разпознатите дялове, например:

```
Partition check:
    hda: hda1 hda2
    hdb: hdb1 hdb2 hdb3
```

Ако поради някаква причина твърдите дискове или дяловете им не бъдат разпознати, няма да можете да ги използвате по никакъв начин.

Ето няколко причини, поради които това може да се случи:

Твърдият диск или контролерът му не се поддържат
: Ако използвате дисков контролер (IDE, SCSI или друг тип), който не се поддържа от Linux, ядрото няма да разпознае дяловете ви при зареждане на системата.

Твърдият диск или контролерът му са конфигурирани неправилно
: Дори ако Linux поддържа дисковия контролер, може да имате проблеми, ако той е неправилно конфигуриран. (Това се случва преди всичко със SCSI контролерите; повечето други контролери работят чудесно и без допълнителни настройки.) Прочетете документацията на твърдия си диск и контролера му, за да получите информация как да решите този тип проблеми. Например твърдите дискове с IDE интерфейс трябва да бъдат настроени с превключвател (jumper), за да работят като "slave" (тоест, като втори диск) на същия контролер). Един лесен начин да проверите дали имате този проблем, е да заредите MS-DOS или друга операционна система, за която знаете че работи с вашия диск и контролера му. Ако имате достъп до диска и контролера му от друга операционна система, следователно проблемът не е в хардуерната конфигурация.

Хардуерните конфликти и начините за разрешаването им са описа-
: ни в предишния раздел "Изолиране на хардуерни проблеми". След-

ващият раздел, "Проблеми със SCSI контролери и устройства", съдържа информация за конфигурирането на SCSI устройства.

Контролерът е конфигуриран правилно, но Linux не го открива
Някои SCSI контролери без BIOS, изискват потребителя да зададе информация за контролера при зареждането на системата. В следващият раздел, "Проблеми със SCSI контролери и устройства", е описано как можете да укажете на ядрото, че в компютъра ви има такива контролери.

Ядрото не разпознава геометрията на твърдия диск
Някои системи като IBM PS/ValuePoint не записват информацията за геометрията на твърдия диск в CMOS паметта си (където Linux очаква да я намери). Освен това, на някои SCSI контролери е необходимо да се укаже къде се намира информацията за геометрията на твърдия диск, за да може Linux да разпознае съществуващите дялове.

Повечето дистрибуции предоставят възможност за ръчно определяне на геометрията на твърдия диск. В общия случай, когато получите поканата на LILO при начално зареждане, можете да укажете геометрията на диска по следния начин:

```
boot: linux hd=цилиндри,глави,сектори
```

където аргументите ***цилиндри, глави*** и ***сектори*** отговарят на броя на цилиндрите, главите и секторите на пътека за вашия твърд диск.

След като инсталирате Linux, можете да инсталирате LILO. LILO е стандартната програма за зареждане на Linux от твърдия диск. По време на инсталирането на LILO можете да зададете геометрията на диска, като по този начин ще избегнете необходимостта да я задавате при всяко зареждане на Linux (виж раздела "Как да използваме LILO" в Глава 5).

Глава 5

Проблеми със SCSI контролери и устройства

Тук ще опишем някои от най-често срещаните проблеми при работа със SCSI контролери и устройства като CD-ROM, твърди дискове и лентови устройства. Ако имате проблеми при разпознаването на диска или контролера си, прочетете този раздел. Нека още веднъж наблегнем на това, че повечето дистрибуции използват модулно ядро и може би ще трябва в началната фаза на инсталационния процес да заредите модула, който поддържа вашия хардуер. Възможно е дистрибуцията ви автоматично да зареди необходимия модул.

[69] SCSI HOWTO

Като допълнение към информацията в този раздел можете да използвате документа Linux SCSI HOWTO. Той съдържа голямо количество полезна информация за SCSI устройствата. В някои случаи конфигурирането на SCSI устройствата е трудно и изисква известни умения.

Възможно е да имате проблеми, ако използвате най-евтините устройства на пазара - например, ако използвате евтини кабели, особено ако имате wide SCSI. Евтините кабели са основния източник на проблеми и могат да са причина за всевъзможни неприятности и главоболия. Ако използвате SCSI устройства, би трябвало да използвате и подходящи кабели.

Ето основните проблеми и препоръки за разрешаването им:

Някое SCSI устройство се открива на всички SCSI идентификатори.

Проблемът възниква, когато устройството използва същия идентификатор (SCSI ID), който използва и контролерът. Трябва да смените настройките на устройството или контролера, така че идентификаторите им да не съвпадат.

Linux извежда съобщение "sense error", дори когато сте сигурни, че устройствата ви са здрави.

Причината за проблема обикновено са лоши кабели или лоша терминация. Ако SCSI шината няма терминатори в двата си края (т.е. не е терминирана), можете да имате проблеми при използване на SCSI устройствата. Освен това, винаги когато имате такива проблеми, трябва да проверите кабелите си. Лошо поставените и кабелите с ниско качество също са източник на подобни проблеми.

Получавате съобщения "timeout error " при работа със SCSI устройст- ва

Обикновено проблемът се предизвиква от IRQ, DMA или I/O конфликт. Проверете дали IRQ сигналът, който се използва от SCSI контролера, е настроен правилно.

Linux не успява да открие SCSI контролер, който използва собствен BIOS

Linux няма да разпознае контролери със собствен BIOS, ако този BIOS е забранен или ако ядрото не разпознае "сигнатурата" на BIOS-а. Повече информация можете да намерите в документа Linux SCSI HOWTO.

Не работят контролерите, които използват "видима " памет на платката си за комуникация с компютъра.

Проблемът възниква, когато тази памет се "кешира" некоректно. Решението е да забраните кеширането на адресното пространство на контролера в CMOS настройките на машината, или, ако това не е възможно, да забраните кеширането на цялата памет.

Когато създавате дялове на SCSI твърд диск, получавате предупреждение "cylinders > 1024" или не можете да заредите Linux от дял, който използва цилиндри с номера по-големи от 1023.

Системният BIOS има ограничение за максималния номер на цилиндър на твърд диск (ограничението е 1023) и затова всеки дял, който използва цилиндри с по-голям номер, не е достъпен за BIOS. Това ограничение не се отнася за Linux. Проблемът възниква от това, че системният BIOS не може да зареди операционна система, която се намира в дял, който започва след 1023-я цилиндър. Съществуват поне две възможни решения на проблема - да заредите Linux като използвате дискета или като използвате дял, който се намира преди 1024-я цилиндър (виж раздела "Как да изберем начина за зареждане на Linux" по-горе в тази глава).

Linux не разпознава SCSI CD-ROM или друго устройство със сменяем носител при зареждането на системата.

Опитайте се да рестартирате компютъра със CD-ROM (или диск) в проблемното устройство - това е изискване на някои устройства.

Ако Linux не разпознае вашия SCSI контролер, може да се наложи изрично да зададете параметрите му по време на стартирането на системата. Това е особено важно за SCSI контролери без собствен BIOS. Повечето дистрибуции ви позволяват при зареждането на Linux от инсталационния носител да зададете номера на IRQ и адреса на паметта на контролера, който използвате. Например, ако използвате SCSI контролер TMC-8xx, можете да въведете:

```
boot: linux tmx8xx=irg,адрес-на-паметта
```

след като получите поканата на LILO при начално зареждане на системата, ***irg*** е номера на IRQ, а ***адрес-на-паметта*** е адреса на видимата памет на контролера. Ако имате проблеми при въвеждането на параметри при зареждането на системата, прочетете документацията на дистрибуцията, която използвате.

Проблеми при инсталиране на софтуера

Ако имате късмет, инсталирането на софтуера ще бъде безпроблемно. Единствените проблеми, с които можете да се сблъскате, би трябвало да са свързани с повреден носител или с липса на място във вашите файлови системи. Следва списък с най-често срещаните проблеми:

Системата извежда съобщение "Read error, fde not found" или друга подобна грешка по време на инсталирането на софтуера.

Тази грешка се получава при проблем с инсталационния носител. Ако инсталирате от дискета, трябва да знаете, че дискетите са доста

чувствителен носител и лесно се повреждат. Използвайте само нови, току-що форматирани дискети. Повечето дистрибуции могат да бъдат инсталирани от Windows дял на твърдия диск. Обикновено това е по-бързо и безпроблемно отколкото използването на дискети.

Ако използвате компакт-диск, проверете дали дискът има драскотини, прах, зацапвания или други подобни проблеми, поради които не може да се чете.

Освен това, причината за проблема може да е неподходящ формат на носителя. Например, когато инсталирате Linux от дискети, много дистрибуции изискват да използвате дискети с висока плътност (high-density) в MS-DOS формат (дискетата за зареждане на Linux е изключение и в повечето случаи не е във MS-DOS формат). Ако не ви остане друг избор, опитайте се да получите нов комплект дискети, или, ако сте изтеглили дистрибуцията си от Интернет, да запишете дискетите отново (използвайте само нови дискети).

Системата извежда съобщения като "tar: read error " или "gzip: not in gzip format ".

Проблемът обикновено се предизвиква от повредени файлове на самия инсталационен носител. С други думи, възможно е дискетата ви да е здрава, но данните на нея да са повредени по някакъв начин. Например, ако сте изтеглили дистрибуцията си от Интернет, като при това сте използвали "текстов" вместо "двоичен" режим, файловете ви ще бъдат повредени и инсталиращият софтуер няма да може да ги използва. Когато използвате FTP, обикновено е достатъчно да изпълните командата *binary,* за да укажете на софтуера, който използвате (FTP-клиента), че файловете трябва да бъдат изтеглени в двоичен режим. Трябва да изпълните тази команда преди да изтеглите самите файлове.

По време на инсталирането си, системата извежда съобщения като "device fidi"

Това е ясен знак, че нямате достатъчно място, за да инсталирате софтуера. Не всяка дистрибуция може да се справи с тази ситуация, така че ако спрете инсталацията, едва ли ще имате работоспособен Linux.

Решението на проблема обикновено е повторно създаване на файловата система с командата *mke2fs,* което ще изтрие частично инсталирания софтуер. След това можете отново да се опитате да инсталирате софтуера, като изберете по-малко пакети. Освен това можете да започнете инсталирането на Linux от самото начало, като отново прецените размерите на дяловете и файловите си системи.

Системата извежда съобщение като "readjntr: 0x10" при работа с твърдия диск.

Обикновено това означава, че твърдият ви диск има лоши блокове. Все пак, ако получите такова съобщение, докато използвате *mkswap* или *mke2fs,* то може да означава, че системата има проблеми при работа с твърдия диск. Това може да е хардуерен проблем (виж раздела "Хардуерни проблеми" по-горе в тази глава) или резултат от лошо зададена геометрия. Например, тази грешка може да се получи, ако при зареждането на системата сте използвали опцията:

```
\\й=цилиндри,глави,сектори
```

за да зададете геометрията на диска си, и данните, които сте задали, не са коректни. Грешката може да се получи и ако геометрията на диска ви не е зададена коректно в CMOS паметта на компютъра.

Системата извежда съобщения от вида "file not found" или "permission denied "

Проблемът може да се получи, ако липсват част от необходимите файлове в инсталационния носител, или има проблем със правата за достъп до файловете. Например, някои дистрибуции на Linux имат грешки в самия инсталиращ софтуер (обикновено тези грешки се поправят бързо и се срещат доста рядко). Ако смятате, че инсталационният софтуер в дистрибуцията ви съдържа грешки и сте сигурни, че не сте допуснали грешка по време на инсталационния процес, можете да се свържете с производителя на дистрибуцията и да съобщите за грешката.

Ако получавате странни съобщения за грешки по време на инсталирането на Linux (особено когато сами сте изтеглили софтуера от Интернет), проверете дали действително имате всички необходими файлове.

Например, за да изтеглят дистрибуция на Linux от някой FTP-сайт, някои хора използват следната FTP-команда:

```
mget *.*
```

Тази команда ще изтегли само файловете, които съдържат символа . в името си; ако съществуват файлове, които не съдържат . в името си, ще ги пропуснете. Правилната команда в този случай е:

```
mget *
```

Най-добрият съвет, който можем да ви дадем, е да проследите стъпките, които сте извършили. Може да мислите, че сте извършили всичко правилно, но в действителност да сте пропуснали малка, но важна стъпка някъде по пътя. В много случаи проблемът може да бъде решен с повторно изтегляне или преинсталиране на софтуера. Не си удряйте главата в стената по-дълго, отколкото е необходимо.

Освен това, ако Linux изведнъж "увисне" по-време на инсталирането на софтуер, вероятно в компютъра ви има някакъв хардуерен проблем (виж раздел "Хардуерни проблеми" по-горе в тази глава).

Проблеми след инсталирането на Linux

Представете си следната ситуация: през целия следобед сте инсталирали Linux. За да освободите място за него, сте премахнали дяловете на Windows и OS/2 и със сълзи на очи сте изтрили SimCity 3000 и Railroad Tycoon 2. Рестартирате компютъра и нищо не се случва. Или дори по-лошо, *нещо* се случва, но не това, което би трябвало да се случи. Какво да правим?

В раздела "Проблеми при зареждане от инсталационния носител" по-горе в тази глава, описахме най-често срещаните проблеми, които възникват при зареждане от инсталационния носител; голяма част от информацията там може да ви помогне. Освен това, може да сте жертва на някой от следните проблеми.

Проблеми при зареждане на Linux от дискета

Ако за зареждане на Linux използвате дискета, при зареждането може да ви се наложи да зададете мястото на твърдия диск, в което се намира основния дял на Linux. Това важи най-вече за случаите, в които за зареждане на Linux използвате самата инсталационна дискета, а не дискета, която е създадена специално за целта.

Когато зареждате от дискета, задръжте натиснат клавиша Shift или клавиша Control. На екрана трябва да се появи покана за въвеждане на команди; натиснете Tab, за да видите списък с предлаганите начини за зареждане на Linux. Например, много дистрибуции ви дават възможност да заредите Linux от дискета като напишете

```
boot: linux root=partition
```

след като получите поканата за въвеждане на команди при начално зареждане, ***partition*** е името на основния дял на Linux, например */dev/hda2*. Инсталационната програма на SuSE Linux предлага възможност за зареждане на току-що инсталирана Linux система, като се използва инсталационната дискета. Детайлите за зареждане на Linux от дискета могат да се открият в документацията на дистрибуцията, която използвате.

Проблеми при зареждане на Linux от твърд диск

Ако сте предпочели да инсталирате LILO, вместо да създадете дискета за зареждане на Linux, би трябвало да можете да заредите Linux директно от твърдия диск. За съжаление, автоматичната процедура за инсталиране на LILO не винаги работи перфектно - възможно е предположенията й за дяловете на диска да са неправилни и в такъв случай трябва отново да инсталирате LILO, за да работи всичко коректно. Инсталирането на LILO е описано в раздела "Как да използваме LILO" в Глава 5.

Глава 5

Следват някои от често срещаните проблеми:

Системата извежда съобщение "Drive not bootable - Please, insert system disk".

Това съобщение се получава, когато по някакъв начин е повреден първият сектор на твърдия диск (т.нар. master boot record - основен запис за начално зареждане). В повечето случаи това не е сериозен проблем и информацията на твърдия ви диск не е повредена. Ето няколко начина да се справите с проблема:

- Възможно е докато сте преразпределяли мястото на твърдия си диск, да сте изтрили дяла, който е бил отбелязан като "активен". MS-DOS и някои други операционни системи се опитват да заредят от "активния" дял при стартиране на системата (в общия случай Linux не обръща внимание на това кой дял е отбелязан като "активен", с изключение на някои дистрибуции на Debian). Едно възможно решение на проблема е да заредите MS-DOS от дискета и след като стартирате FDISK, да отбележите дяла на MS-DOS като активен.

  Освен това можете да опитате да изпълните следната команда (ако разполагате с MS-DOS 5.0 или по-нова версия):

  ```
  FDISK /MBR
  ```

  Тази команда ще се опита да създаде master boot record, който ще изтрие LILO, но ще може да зареди MS-DOS. Ако нямате MS-DOS на твърдия си диск, трябва да заредите Linux от дискета и да се опитате след това да инсталирате LILO.

- Проблемът може да се получи, ако сте създали дял на MS-DOS с Linux-версията на *fdisk* (или обратното). Би трябвало да създавате дялове на MS-DOS само с MS-DOS-версията на FDISK (същото се отнася и за всяка друга операционна система). Най-доброто решение е или да започнете от самото начало и да създадете дяловете на диска коректно, или просто да изтриете и отново да създадете засегнатите дялове с коректната версия на *fdisk*.

- Възможно е да се е провалила инсталационната процедура на LILO. В такъв случай трябва да заредите Linux от дискетата за зареждане на Linux (ако имате такава дискета) или от оригиналния инсталационен носител. И в двата случая имате възможност при зареждане да зададете основния дял на инсталацията на Linux. За целта, при зареждане задръжте натиснат клавиша Shift или клавиша Control и след като получите поканата на LILO за въвеждане на команди, натиснете Tab, за да видите списък с възможните начини за зареждане на Linux.

Когато зареждате от твърдия диск, вместо Linux тръгва MS-DOS (или друга операционна система).

На първо място се уверете, че действително сте инсталирали LILO по време на инсталирането на Linux. Ако не сте, компютърът ще продължи да зарежда MS-DOS (или операционната система, която сте имали преди да инсталирате Linux) всеки път, когато се опитате да заредите операционна система от твърдия диск. За да заредите Linux от твърдия си диск, трябва да инсталирате LILO (виж раздела "Как да използваме LILO" в Глава 5).

Глава 5

От друга страна, ако *сте* инсталирали LILO и се зарежда друга операционна система вместо Linux, това означава, че конфигурацията на LILO е по подразбиране да стартира другата операционна система. Докато системата се зарежда, задръжте натиснат клавиша Shift или Control и след като получите поканата на LILO за въвеждане на команди, натиснете Tab. Би трябвало да получите списък с операционните системи, които могат да бъдат заредени от LILO; просто изберете подходящата опция (обикновено това е просто **linux),** за да заредите Linux.

Ако искате Linux да бъде операционната система, която се зарежда по подразбиране, трябва да отново да инсталирате LILO.

Възможно е да сте се опитали да инсталирате LILO, но инсталационната процедура да се е провалила по някаква причина (виж предишната точка).

Проблеми при влизане в системата

След като заредите Linux, би трябвало да получите покана за влизане в системата:

```
Linux login:
```

Можете да получите инструкции какво да правите по-нататък от документацията на дистрибуцията, която използвате, или от самата работеща система. При много дистрибуции трябва да влезете в системата като

root, без парола. Други възможни потребителски имена, които можете да опитате, са **guest** и **test.**

Повечето дистрибуции на Linux ви питат за паролата на **root,** която сте въвели по време на инсталирането на системата. Спомнете си каква парола сте написали и да я въведете отново. Ако вашата дистрибуция не ви е питала за паролата на потребителя **root,** опитайте се да използвате "празна" парола (просто натиснете Enter).

Ако не можете да влезете в системата, прочетете документацията на дистрибуцията си; може би някъде ще намерите потребителското име и паролата, които трябва да използвате. Възможно е тези данни да са ви били предоставени по време на инсталирането на Linux или да се отпечатват на екрана преди първото влизане в системата.

Една от възможните причини за проблема е грешка при инсталирането на инициализационните и конфигурационните файлове на Linux. В този случай би трябвало да инсталирате отново (поне част от) софтуера, или да заредите Linux от инсталационния носител и да се опитате да решите проблема ръчно.

Проблеми при използването на системата

Ако успеете да влезете в системата, би трябвало автоматично да се стартира командния интерпретатор и да получите покана за въвеждане на команди (например # или $), след което можете свободно да разгледате системата. Следващата стъпка в този случай е да опитате някои от процедурите, описани в Глава 4. Все пак, понякога възникват проблеми при началното използване на системата.

Глава 4

Най-често срещаният проблем в началното конфигуриране на системата са некоректни права за достъп до някои файлове и директории. Като резултат можете да получите съобщението:

```
Shell-init: permission denied
```

веднага, след като влезете в системата. (Всъщност, всеки път, когато видите съобщението permission denied, можете да сте абсолютно сигурни, че проблемът е свързан с правата за достъп.)

В повечето случаи решението е да използвате командата *chmod,* за да поправите правата за достъп до съответните файлове или директории. Например, някои стари дистрибуции на Linux задават неправилни права за достъп (0644) до основната директория. Проблемът може да бъде отстранен, ако влезете в системата като **root** и въведете командата:

```
# chmod 755 /
```

(Правата за достъп до файловете са описани в раздела "Собственост и права за достъп до файл" в Глава 4.) За да изпълните горната команда, трябва да заредите Linux от инсталационния носител и да монтирате основната файлова система ръчно - доста тежка задача за повечето начинаещи потребители.

Глава 4

Докато използвате системата, можете да имате проблеми с некоректно настроени права за някои файлове и директории, софтуерът да не работи по начина, по който е конфигуриран и т.н. Добре дошли в света на Linux! Макар че повечето дистрибуции са доста добре направени, не трябва да очаквате, че са перфектни. Не бихме искали тук да описваме всички възможни проблеми. Вместо това, в тази книга ще ви помогнем да решите много от тези конфигурационни проблеми, като ви научим как сами да ги откривате и поправяте. Тази философия е описана донякъде в Глава 1. В Глава 5 ви даваме съвети как да решите голяма част от общите конфигурационни проблеми.

Глава 1
Глава 5

ГЛАВА 4

Основни команди и концепции в UNIX

Ако досега сте използвали MS-DOS или друга, различна от Unix операционна система, ще ви се наложи да научите много нови неща. Ще бъдем откровени - Unix е един различен свят.

В тази глава ще опишем основите на Unix за читателите, които никога не са работили с тази операционна система. Ако идвате от света на MS-DOS, Microsoft Windows или друга среда, информацията в тази глава ще бъде от изключителна важност за вас. За разлика от други операционни системи, работата с Unix не е интуитивна. Много от командите изглеждат така, сякаш имат случайни имена или синтаксис. Причината за това е възрастта на системата - операционната система Unix е създадена преди много години. Освен това, макар че много от командите изглеждат подобни на съответните команди в MS-DOS, между тях съществуват важни разлики.

Съществуват десетки други книги, които описват основните положения при работата с Unix. Би трябвало по рафтовете на компютърния щанд във всяка голяма книжарница да намерите поне три-четири книги за Unix (някои от книгите, които харесваме, са изброени в Библиографията). Все пак трябва да отбележим, че повечето от тези книги описват Unix от гледната точка на потребител, който седи пред работна станция или терминал, свързан към голяма машина, а не от гледната точка на ентусиаст, който използва собствена Unix система на личния си компютър.

Освен това, често тези книги описват подробно рутинните аспекти на работата с Unix, например отегчителни команди за манипулиране на текстови файлове като *awk, tr* и *sed,* повечето от които никога няма да са ви необходими, освен ако не решите сериозно да се възползвате от тън-

костите при работата с Unix. Всъщност, в много от книгите за Unix се говори за оригиналния редактор на редове *ed,* който отдавна е заменен от пълноекранните текстови редактори ***vi*** и Emacs. Макар че много от книгите за Unix, които можете да намерите в днешни дни, съдържат голямото количество полезна информация, в немалко от тях ще намерите цели глави със скучен материал, който не ви е необходим в момента.

Вместо да навлизаме в джунглата на манипулирането на текстови файлове, синтаксиса на командния интерпретатор и други теми, в тази глава се опитваме да опишем основните команди, които са ви необходими, за да започнете да работите със системата, ако идвате от различна от Unix среда. Тази глава далеч не е достатъчна; за едно истинско ръководство за начинаещи потребители на Unix е необходима цяла книга. Надяваме се, че тук ще намерите достатъчно информация, за да оцелеете в началото на приключенията си с Linux, и че ще инвестирате в някоя от споменатите книги за Unix, когато почувствате необходимост. Ще ви дадем достатъчно познания за основите на Unix, за да можете да използвате терминала си, да проследявате задачите и да въвеждате основните команди на системата.


Глава 5

Глава 5, *Основи на системната администрация,* съдържа информация за системната администрация и поддръжка. Тази глава е от изключителна важност за всеки, който използва собствена Linux система. Дори да нямате никакъв опит с Unix, би трябвало лесно да се справите с материала в Глава 5, като използвате ръководството в тази глава.


Глава 9

Тук няма да се спираме на една много голяма тема - редактирането на текстови файлове. Това е едно от първите неща, които трябва да научите, за да можете да работите с някоя операционна система. Двата най-популярни текстови редактори за Linux - ***vi*** и Emacs - са описани в началото на Глава 9, *Редактори, текстови инструменти, графика и печат.*

Влизане в системата

Ще приемем, че инсталирането е минало без никакви проблеми и на екрана ви е изписана следната покана за влизане в системата:

```
Linux login:
```

Много потребители на Linux нямат този късмет - налага им се ръчно да извършат някои трудни поправки, докато системата все още не е конфигурирана или е в еднопотребителски режим. Но засега ще опишем влизането във функционираща Linux система.

Разбира се, влизането в системата разграничава потребителите един от друг. То позволява много хора да работят на една и съща система и га-

рантира, че вие сте единствения човек, който има достъп до вашите файлове.

Вероятно сте инсталирали Linux вкъщи и сега си мислите "Голяма работа. Никой освен мен не използва този компютър и скоро няма да ми се наложи да влизам в системата". Но влизането с личния ви account осигурява и известна защита - вашият account ме може да повреди или изтрие важни за работата на Linux файлове. За такива деликатни цели трябва да използвате account-а на системния администратор (виж следващата глава).

Ако свързвате компютъра си към Интернет (дори и с модем), не използвайте тривиални пароли в системата си. Паролите на потребителите в системата ви трябва да бъдат колкото е възможно по-сложни - да включват букви и препинателни знаци и да не са имена или обикновени думи.

Вероятно докато сте инсталирали Linux, инсталационната програма ви е дала възможност да създадете свой account. Ако сте създали такъв account, когато получите поканата за влизане в системата, трябва да напишете потребителското име, което сте избрали. Ако все още нямате account, трябва да използвате потребителското име root, понеже със сигурност съществува такъв account. Някои дистрибуции създават допълнителен account с потребителско име install или някое друго име, който да използвате докато инсталирате системата.

След като въведете потребителското име на вашия account, би трябвало да получите покана за въвеждане на паролата:

```
Password:
```

Трябва да въведете паролата си. За тази операция терминалът изключва нормалното отпечатване на екрана на символите, които въвеждате, затова никой няма да може да прочете паролата ви, ако наблюдава монитора, докато я пишете. Ако не получите поканата за въвеждане на парола, това означава, че не използвате парола. Настоятелно ви съветваме да използвате парола, за да се защитите от злонамерена намеса на други хора; по-долу ще обсъдим този въпрос подробно.

Между другото, в името и паролата се прави разлика между малките и главните букви. Проверете дали не свети индикаторът Caps Lock на клавиатурата ви, защото ако въведете **ROOT** вместо `root`, няма да влезете в системата.

След като влезете в системата, ще видите покана за въвеждане на команди. Ако сте влезли като root, това може да е просто:

```
#
```

За останалите потребители поканата за въвеждане на команди обикновено е символа $. Възможно е поканата да съдържа и името, което сте задали на системата си или текущата директория. Независимо от конкретния вид на поканата, която сте получили, сега можете да въвеждате команди. За такива случаи ще казваме, че се намирате "в командния интерпретатор" (shell level), а поканата, която виждате, ще наричаме покана на командния интерпретатор (shell prompt). Използваме тези термини, защото в момента работи програма, наречена команден интерпретатор (shell), която обработва командите ви. Точно в този момент можем да игнорираме командния интерпретатор, но по-долу в тази глава ще видим, че той върши немалко полезни неща за нас.

Когато показваме команди в тази глава, ще отбелязваме поканата за въвеждане на команда просто като $. Например, ако в текста срещнете:

```
$ pwd
```

това означава, че командният интерпретатор е отпечатал $ и се очаква вие да въведете `pwd`.

Задаване на парола

Ако все още нямате парола, можете лесно да я зададете - достатъчно е просто да изпълните командата *passwd*. Тази команда ще ви попита за паролата ви, след което ще ви помоли да я въведете още веднъж, за да е сигурно, че не сте допуснали грешка при първото й въвеждане.

Съществуват няколко стандартни препоръки за начина, по който трябва да изберете паролата си, така че да е трудно друг човек да се досети каква е тя. Някои системи дори проверяват паролата ви и я отхвърлят, ако тя не отговаря на минималните критерии. Например, често се казва, че паролата трябва да се състои поне от 6 символа. Освен това, в паролата би трябвало да има смесица от главни и малки букви или да се използват символи, които не са букви или цифри.

За да смените паролата си, достатъчно е отново да въведете командата *passwd*. Командата ще ви попита за старата парола (за да е сигурно, че сте вие), след което ще можете да смените паролата си по описания по-горе начин.

Виртуални конзоли

Като многозадачна система Linux ви предоставя множество интересни начини да извършвате едновременно няколко неща. Можете да започнете инсталиране на софтуер, което продължава много време, и без да изчаквате то да завърши, да превключите на програма за четене на елект-

ронната поща или да започнете компилация на програма. Всички тези процеси ще работят едновременно. Това е основната причина, поради която много потребители на MS-DOS харесват Linux (макар че последните версии на Microsoft Windows най-после овладяха многозадачната работа).

Повечето потребители на Linux могат да използват системата X Window, когато искат да извършат асинхронно няколко задачи. Но преди да успеете да пуснете X, можете да направите нещо подобно като използвате виртуалните конзоли. Тази възможност съществува и в някои други версии на Unix, но не е универсална.

За да опитате как работят виртуалните конзоли, задръжте натиснат левия клавиш Alt и натиснете някой от функционалните клавиши от F1 до F8. Всеки път, когато натискате функционален клавиш, ще виждате изцяло различен екран, в който има покана за влизане в системата. Можете да влезете в системата от различни виртуални конзоли, все едно сте двама различни потребителя и като сменяте конзолите, да извършвате различни действия във всяка от тях. Възможно е дори да стартирате независима сесия на X във всяка конзола. По подразбиране, системата X Window използва виртуална конзола с номер 7. Следователно, ако стартирате X и след това се прехвърлите на някоя от виртуалните конзоли с текстов интерфейс, бихте могли да се върнете в X като натиснете AK-F7. Ако във вашата система комбинацията "Alt + функционален клавиш" извежда менюто на X или има някаква друга функция, за превключване към виртуалните конзоли можете да използвате комбинацията "Ctrl + Alt + функционален клавиш".

В по-ранните версии на Linux (до ядро с версия 1.1.54), броят на достъпните виртуални конзоли беше фиксиран, но можеше да се променя с поправка, прекомпилиране и ново инсталиране на ядрото; по подразбиране се поддържаха 8 виртуални конзоли. Със съвременните версии на ядрото Linux може при необходимост "в движение" да създаде нова виртуална конзола. Все пак, това не означава, че можете просто да превключите на виртуална конзола номер 13 и да влезете в системата. Можете да влезете в системата само във виртуална конзола, за която има работещ процес *getty* (виж следващата глава за повече информация).

Често използвани команди

Броят на командите в една типична Unix система е достатъчен, за да изпълни няколкостотин справочни страници. Освен това, всеки може да добавя нови команди. Тук ще опишем само команди, с които да можете да се ориентирате в системата и да разгледате какво има в нея.

Директории

Както при MS-DOS и практически всяка съвременна компютърна система, файловете в Unix са организирани в йерархична структура от директории. Unix не налага правила за това, къде трябва да бъдат файловете, но през годините са създадени няколко конвенции. Затова обикновено в Linux ще откриете директория */home,* в която се съхраняват потребителските файловете. За всеки потребител съществува отделна поддиректория в йерархията */home.* Например, ако вашето потребителско име е **mdw,** файловете ви ще се намират в директорията */home/mdw.* Това е вашата лична директория (home directory). Разбира се, в нея можете да създавате собствени поддиректории.

Както виждате, отделните части в името на директорията се отделят с наклонени черти. Често за означаване на тази последователност от разделени с наклонена черта имена се използва терминът *път* или *име на път*.

А в коя директория е директорията */home?* Разбира се, в директорията с име /- така наречената основна директория. Вече споменахме основната директория, когато говорихме за монтирането на файловите системи.

Когато влезете в системата, започвате работа във вашата лична директория. Можете да проверите това, като използвате командата "print working directory" (отпечатай работната директория) или *pwd:*

```
$ pwd
/home/mdw
```

Системата потвърждава, че сте в директорията */home/mdw*.

Едва ли ще ви бъде много интересно, ако стоите само в една директория през цялото време. Можете да използвате командата *ed,* за да се преместите в друга директория:

```
S ed /usr/bin
S pwd
/usr/bin
S ed
```

Къде се намирате в този момент? Ако въведете командата *ed* без аргументи, ще се върнете във вашата лична директория. Между другото, пълното й име често се заменя от символа тилда (~). Например низът *~/programs* означава, че директорията *programs* се намира във вашата лична директория.

Докато говорим за това, да направим нова директория с име *-/programs.* От личната си директория можете да изпълните командата:

```
$ mkdir programs
```

или, ако използвате пълен път:

S mkdir /home/mdw/programs

Влезте в тази директория:

$ cd programs
$ pwd
/home/mdw/programs

Специалната последователност от символи .. се използва за означаване на "директорията, която се намира точно над текущата в йерархията на директориите". Така че, можете да се върнете в личната си директория като въведете командата:

```
$ cd . .
```

Обратната команда на *mkdir* е *rmdir* и се използва за изтриване на директории:

$ rmdir programs

Глава 9

Аналогично, командата ***гт*** изтрива файлове. Тук не показваме как да я използвате, защото все още не знаете как се създават файлове. Можете да създадете файл с текстовите редактори *vi* и Emacs (виж Глава 9) или с някоя от командите, които ще опишем по-късно в тази глава. С опцията ***-г*** (рекурсивно), командата ***гт*** ще изтрие цялата директория и всичките й поддиректории. (Използвайте тази възможност много внимателно!)

Как да видим списък с файловете в дадена директория

Въведете командата *Is,* за да видите какво има в дадена директория. Ако въведете *Is* без аргументи, командата ще изведе съдържанието на текущата директория. Можете да добавите като аргумент името на друга директория, за да видите съдържанието й:

```
$ Is /home
```

Някои системи използват украсен вариант на *Is,* който показва специалните файлове - като директории и изпълними файлове - с удебелен шрифт, а понякога дори в различен цвят. Ако искате да промените цветовете по подразбиране, трябва да редактирате файла */etc/DIRCOLORS,* или да създадете копие от този файл с име *.dirjoolors* в личната си директория и след това да го редактирате.

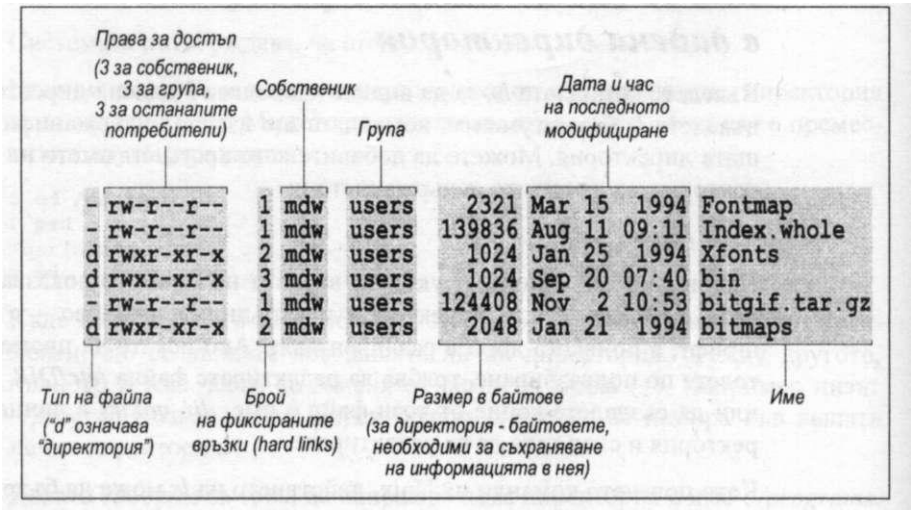
Като повечето команди на Unix, действието на *Is* може да бъде променено с опции, които започват с тире (-). Преди тирето трябва да оставите интервал. Една полезна опция на *Is* е *-a* (идва от "all" - всички). Тази опция ще ви открие богатства, които едва ли сте очаквали да намерите в личната си директория:

```
$ cd
$ ls -a
                          .bashrc                  .fvwmrc
                          .emacs                   .xinitrc
.bash_history             .exrc
```

Единичната точка задава текущата директория, а двете точки задават директорията, която се намира точно над текущата. Но какви са останалите файлове, които започват с точка? Те се наричат "скрити" файлове. Ако името на файл започва с точка, то не се извежда при изпълнение на "обикновената" команда ***ls.*** Много програми използват скрити файлове за съхранение на настройките си - тоест на начина, по който вие бихте искали да работят тези програми. Например, можете да добавите свои команди във файла ***.Xdefaults,*** за да измените начина по който работи системата X Window. През повечето време не бихте искали да виждате тези файлове, но те са много полезни, когато конфигурирате системата си. По-късно ще опишем някои от тях.

Друга полезна опция на ***ls*** е -/ (идва от "long" - дълго). С тази опция ***ls*** показва допълнителна информация за файловете. Фигура 4-1 съдържа типичен резултат от изпълнението на командата ***ls -l*** и описание на смисъла на различните полета.

Полетата, които показват правата за достъп, собственика и групата, са описани по-долу в тази глава в раздела "Собственост и права за достъп до файл". Командата ***ls*** показва също размера на файловете и кога за последен път са били променяни.



Фигура 4-1: Резултат от изпълнението на командата ls -l

Разглеждане на файлове

Един от начините да разгледате съдържанието на даден файл е да използвате текстов редактор, например:

$ emacs .bashrc

Но ако искате само набързо да прегледате съдържанието на файла без да го редактирате, по-удобно е да използвате други команди. Най-простият начин е да използвате командата със странното име *cat* (името идва от глагола concatenate, "свързвам", защото командата може да се използва и за свързване на няколко файла в един):

$ cat .bashrc

Но един дълъг файл ще премине твърде бързо през екрана на компютъра и няма да можете да го разгледате. Затова вместо *cat* по-често се използва командата *more:*

$ more .bashrc

На екрана се извежда част от съдържанието на файла; можете да я разгледате и след това да натиснете клавиша интервал, за да видите следващата част. *more* разполага с голямо количество полезни опции. Например, можете да потърсите текстов низ във файла - натиснете клавиша / (наклонена черта), напишете текста и натиснете Enter.

Популярен вариант на *more* е командата *less.* Тя има дори по-големи възможности. Например, можете да отбележите дадено място във файла и да се върнете на него по-късно.

Символни връзки

Понякога е удобно да съхранявате определен файл на едно място и да симулирате, че той се намира на друго. Това се прави най-често от системните администратори, а не на обикновените потребители. Например, ако имате инсталирани няколко версии на една и съща програма, наречени *prog.0.9, prog. 1.1* и т.н., можете да използвате командата *prog* за стартиране на тази версия, с която работите в момента. Друг пример - понякога се налага да инсталирате някой файл в даден дял просто защото в този дял има свободно място, и да симулирате, че файла се намира в друг дял, където очаква да го намери даден софтуер.

Unix предлага така наречените *връзки,* за да ви помогне в такива ситуации. В този раздел ще разгледаме най-гъвкавия и разпространен тип връзки - *символните връзки.* Символната връзка е нещо като празен файл, който просто сочи към друг файл. Когато редактирате, четете или изпълнявате файл, който представлява символна връзка, системата ще използва истинския файл, към който сочи тази връзка. Символните връз-

ки работят до голяма степен като така наречените "shortcuts" в Windows 95/98, но са значително по-мощно средство.

Да се върнем на примера с *prog.* Искате да създадете връзка с име *prog,* която сочи към истинския файл, наречен *prog. 1.1.* Въведете командата:

```
S In -s prog.1.1 prog
```

По този начин създадохте нов файл с име *prog,* който в известен смисъл е празен файл; когато го стартирате, системата всъщност стартира *prog. 1.1.* Да видим какво ще каже *Is —I* за символната връзка:

```
$ Is -1 prog
lrwxrwxrwx   2 mdw     users        8 Nov 17 14:35 prog -> prog.1.1
```

Буквата l в началото на реда означава, че файлът е връзка, а низа -> показва истинския файл, към който сочи връзката.

Ще се убедите колко лесно се работи със символни връзки, след като разберете идеята за това един файл да сочи към друг файл. Ще срещате връзки всеки път, когато инсталирате софтуерни пакети.

Командни интерпретатори

Както казахме по-горе, след като влезете в системата, управлението се поема от команден интерпретатор (или от графичен интерфейс, ако системата ви е конфигурирана да използва display manager). Същото се получава, когато отворите прозорец на *xtenn* под X. Командният интерпретатор анализира и изпълнява вашите команди. Преди да продължим нататък, ще опишем накратко различните видове командни интерпретатори, защото тези видове до известна степен определят част от следващия материал.

Може би изглежда обърквашц фактът, че Linux предлага много и различни командни интерпретатори. Просто приемете това като резултат от еволюцията на операционната система. Повярвайте ни, не бихте искали да работите с първия команден интерпретатор за Unix - Bourne shell. Макар че е бил много мощен потребителски интерфейс за времето, в което е създаден (средата на 70-те години), той няма голяма част от полезните интерактивни възможности на съвременните командни интерпретатори, включително и част от възможностите, описани в този раздел. С течение на времето са разработени много други командни интерпретатори, така че днес можете да изберете този, който най-добре отговаря на

Програма, която ви позволява да влезете в системата директно в графичен режим - б.пр.

начина ви на работа. Някои от командните интерпретатори, които можете да използвате под Linux са:

bash
: Bourne Again shell. Най-често използваният (и най-мощен) команден интерпретатор за Linux. Съвместим е с Bourne shell, отговаря на стандарта POSIX, създаден е и се разпространява от Free Software Foundation (проекта GNU). Предлага редактиране на командния ред, история на използваните команди и съвместимост с класическия Bourne Shell.

csh
: C shell. Разработен е в Бъркли. Практически съвместим с Bourne shell при интерактивна употреба, но има съвсем различен интерфейс при програмиране. Не предлага редактиране на командния ред, но има много мощна алтернатива - история на използваните команди.

ksh
: Korn shell. Може би най-популярният команден интерпретатор в Unix системите по света и първият, който въвежда модерните техники при работа с Bourne shell (включително и някои техники, заимствани от C shell). Съвместим с Bourne shell. Предлага редактиране на командния ред.

sh Bourne shell. Оригиналният команден интерпретатор. Не предлага редактиране на командния ред.

tcsh
: Подобрен C shell. Предлага редактиране на командния ред.

zsh
: **Z** shell. Най-новият команден интерпретатор. Съвместим с Bourne shell. Предлага редактиране на командния ред.

Въведете следващата команда, за да откриете какъв команден интерпретатор използвате. Командата отпечатва пълното име (включително директорията, в която се намира) на командният интерпретатор. Не забравяйте да въведете символа $:

```
$ echo $SHELL
```

Най-вероятно използвате *bash* (Bourne Again shell). Ако използвате някой друг команден интерпретатор, препоръчваме ви да го смените с *bash*. Той е мощен, отговаря на стандарта POSIX, има добра поддръжка и е много популярен в Linux. Можете да смените командния си интерпретатор с командата *chsh:*

$ chsh

```
Enter password: Напишете паролата си - прави се с цел сигурност
Changing the login shell for kalle
Enter the new value, or press return for the default
        Login Shell [/bin/sh]:/bin/bash
```

За да може обикновеният потребител да избере някой конкретен команден интерпретатор, този интерпретатор трябва да е инсталиран и системният администратор трябва да го опише във файла */etc/shells*.

Съществуват няколко начина за описание на основните разлики между командните интерпретатори. Единият е всички командни интерпретатори да се разделят на съвместими с Bourne shell и съвместими със *csh*. Това деление представлява интерес за тези, които пишат програми за командния интерпретатор (тези програми са известни като скриптове на командния интерпретатор или shell scripts). Bourne shell и C shell използват различни програмни конструкции. Днес повечето хора смятат, че командните интерпретатори, съвместими с Bourne shell, са по-добри и съществуват много програми за Unix, които работят само с Bourne shell.

Друг начин за категоризиране на командните интерпретатори е да^се отделят тези, които предлагат редактиране на командния ред (всички съвременни командни интерпретатори), от тези, които не предлагат, *sh* и *csh* не предлагат тази полезна възможност.

Като комбинираме двата критерия - командните интерпретатори да бъдат съвместими с Bourne shell и да предлагат редактиране на командния ред - най-добри се оказват *bash, ksh* и *zsh*. Опитайте различните командни интерпретатори преди да направите своя избор. Би било добре да можете да работите с повече от един команден интерпретатор - в случай че един ден ви се наложи да работите на система с ограничен избор на командни интерпретатори.

Полезни клавишни комбинации и как да ги накараме да работят

Когато въвеждате команда, използвайте клавиша Backspace, за да изтриете последния въведен символ. Клавишната комбинация Ctrl-U изтрива целия ред. Когато въведете дадена команда и я стартирате, можете да прекъснете изпълнението й с Ctrl-C или да го спрете временно с Ctrl-Z (изпълнението може да се възстанови с командата^ - от "foreground").

* Ctrl-U се въвежда като натиснете клавиша Ctrl и след това, без да го пускате, натиснете клавиша U.

Ако някой от изброените клавиши не работи, значи терминалът ви не е настроен коректно. Можете да решите проблема с командата *stty*. Използвайте следния синтаксис:

```
stty функция клавиш
```

където ***функция*** е действието, което искате да се извърши, а ***клавиш*** е клавишната комбинация, която натискате. Комбинациите, които включват Ctrl, се означават със символа * преди името на клавиша.

Следват няколко примерни команди, с които можете да настроите изброените по-горе функции:

```
$ stty erase ^H
$ stty kill  ^U
$ stty intr  ^C
$ stty susp  *Z
```

Комбинацията "н съответства на ASCII кода, който се генерира от клавиша Backspace.

Между другото, командата *stty -a* отпечатва списък с текущите настройки на терминала. Не е задължително да разбирате всички детайли в този списък - *stty* е сложна команда и може да се използва по много начини, някои от които изискват задълбочени познания за терминалите.

Улеснения при въвеждането на команди

Ако сте изпробвали всички примери в тази глава, вероятно ви е омръзнало да пишете отново и отново едни и същи команди. Особено неприятно е когато допуснете грешка и трябва отново да въведете цялата команда. Тук е мястото, където командният интерпретатор може значително да ви улесни. Той не прави работата с Unix проста като при интерфейсите от типа посочи-и-щракни, но може да ви помогне да работите значително по-бързо в командна среда.

В този раздел ще опишем редактирането на командния ред. Съветите, които ви даваме, работят с *bash, ksh, tcsh* и *zsh*. Командните интерпретатори, предлагащи редактиране на командния ред, третират последните (около) 50 реда, които сте въвели, като буфер в текстов редактор. Можете да ги разглеждате и редактирате по начина, по който редактирате документ. След като натиснете клавиша Enter, командният интерпретатор ще изпълни командите в текущия ред.

Автоматично довършване на думи

Нека започнем с нещо просто, което ще ви спести много време. Въведете следния текст, без да натискате клавиша Enter:

```
$ cd /usr/inc
```

Сега натиснете клавиша Tab. Командният интерпретатор ще добави текста **lude** в края на реда, като по този начин довършва името на директорията */usr/include*. Сега можете да изпълните командата като натиснете клавиша Enter.

Критерият при задаване на име на файл е "минимално разграничение". Когато въвеждате име на файл, достатъчно е да въведете само началните му символи, за да го разграничите от останалите файлове в същата директория. Командният интерпретатор може да открие неговото име и да го довърши; ако името е на директория, автоматично ще се добави наклонена черта след името й.

Същата възможност на командния интерпретатор се използва и за въвеждане на команди. Например, ако въведете:

```
$ ета
```

и натиснете клавиша Tab, командният интерпретатор автоматично ще добави **es,** за да се получи **emacs** (освен ако някоя друга команда не започва **с** ета).

Какво ще стане, ако съществуват няколко имена, които започват със символите, които сте въвели? В такъв случай командният интерпретатор ще допълни текста до мястото, след което имената се различават. Повечето командни интерпретатори спират дотук, *bash* предлага по-елегантно решение: ако натиснете още веднъж Tab, ще бъде отпечатан списък с всички възможни имена. Например, ако въведете:

```
$ cd /usr/l
```

и натиснете два пъти клавиша Tab, *bash* ще отпечата следното:

```
lib         local
```

Придвижване между въведените команди

Натиснете стрелка нагоре, за да видите последната въведена команда. Можете да използвате стрелка нагоре и стрелка надолу, за да се придвижвате между въведените команди. Ако искате да смените символ в текущия ред, можете да използвате стрелка наляво и стрелка надясно.

Да предположим, че се опитвате да изпълните следната команда:

```
$ mroe .bashrc
bash: mroe: command not found
```

Това е елементарна грешка - написали сте **mroe** вместо **more.** За да поправите грешката, натиснете стрелка нагоре (за да редактирате последния въведен ред), след което натиснете няколко пъти стрелка наляво, докато курсорът застане над буквата о в **mroe.** Бихте могли да изтриете буквите **о** и **r** с клавиша Backspace, след което да ги въведете правилно. Съществува дори по-добра възможност - просто натиснете Ctrl-T. Тази команда сменя местата на **о** и г, след което можете натиснете Enter, за да изпълните командата.

Съществуват още много клавишни комбинации за редактиране на командния ред. Но дори само основните комбинации, които ви показахме тук, би трябвало много да ви помогнат. Ако се научите да работите с текстовия редактор Emacs, ще установите, че повечето от клавишните комбинации в него работят и в командния интерпретатор. А ако сте фен на *vi,* можете да настроите командния интерпретатор, така че да използва клавишните комбинации на *vi,* вместо тези на Emacs. В *bash, ksh* и *zsh* това се прави с командата:

```
$ export VISUAL=vi
```

В *tcsh* трябва да въведете:

```
$ setenv VISUAL vi
```

Използване на маски в имената на файловете

Друг начин да спестите време, когато въвеждате команди, е да съкратите имената на файловете със специални символи. Обикновено тези символи се използват за едновременно обозначаване на няколко файла. Понякога тази функция на командния интерпретатор се нарича globbing (обобщаване).

Тази функция до известна степен е реализирана в MS-DOS - можете да използвате символа ?, за да обозначите "произволен символ", или *, ако искате да обозначите "произволен низ от символи". Unix предлага и тази функционалност, но реализацията е значително по-елегантна и с по-големи възможности.

Да предположим, че имате директория, която съдържа следните файлове с изходен код на С:

```
$ Is
inv1jig.c   inv2jig.c   inv3jig.c   invinitjig.c   invpar.c
```

За да изведете списък с файловете, които съдържат цифра в името си, можете да въведете следната команда:

```
$ Is inv?jig.c
invljig.c   inv2jig.c   inv3jig.c
```

Командният интерпретатор се опитва да замени символа ? точно с един символ. Затова се отпечатват файловете *invljig.c, invìjig.c* и *inv3jig.c,* но не и *invinitjig.c,* защото след inv следват 4 символа преди низа jig.c.

Ако не се интересувате от втория файл, можете да зададете файловете, които искате, като използвате квадратни скоби:

```
$ Is inv[13]jig.c
invljig.c   inv3jig.c
```

Ако някой от символите в квадратните скоби съвпада със съответния символ в името на файла, критерият е изпълнен и файлът се отпечатва. Освен това е възможно в квадратните скоби да зададете всички символи в някакъв интервал:

```
$ Is inv[l-3]jig.c
invljig.c   inv2jig.c   inv3jig.c
```

По този начин се извеждат отново и трите файла. Критерият е "всички символи от 1 до 3 включително". Бихте могли да зададете например всички цифри с критерия `[0-9]` или всички букви с `[a-zA-z]`. Във втория случай трябва да се изброят малките и главните букви, защото командния интерпретатор ги различава. Между другото, наредбата на символите се определя от символната таблица ASCII.

Да предположим, че искате да видите и инициализационния файл. Можете да използвате символа *, защото искате на критерия да отговарят всички имена на файлове, които започват с inv и завършват на jig, без значение колко символа има между тези два низа:

```
$ Is inv*jig.c
invljig.c   inv2jig.c   inv3jig.c   invinitjig.c
```

Звездата в маската всъщност означава "нула или повече символа", затова ако в директорията има файл с име *invjig.c,* той също ще бъде изведен.

За разлика от MS-DOS, командните интерпретатори на Unix ви позволяват да комбинирате специални и обикновени символи по произволен начин. Като пример ще изведем списък, съдържащ всички файлове с изходен код на C (имената им завършват с *.c)* и всички файлове с обектен код (имената им завършват с .o), при допълнителното условие имената на файловете да съдържат една цифра. Маската, която ще използваме, обобщава всички примери, които разгледахме в този раздел:

```
$ Is *[0-9]*. [co]
```

Маските са много полезни при писането на програми за командния интерпретатор (така наречените shell-скриптове), защото в много случаи програмистът не знае точно колко файла трябва да се обработят. Например, можете да използвате израза *log*,* когато пишете скрипт за обработване на всички log-файлове с имена *logOOl, logOOl* и т.н. Независимо от точния им брой, израза *log** връща всички такива файлове.

Едно последно предупреждение: маските в имената на файловете не са регулярни изрази. Много програми използват регулярни изрази за задаване на групи от текстови низове. Регулярните изрази са извън обхвата на тази книга, но можете да намерите тяхно описание в много книги за приложения за Unix. Ще намерите няколко примера за регулярни изрази в Глава 13, *Езици за програмиране.*

Глава 13

Съхраняване на изведените съобщения

Системните администратори (а също и немалко други хора) виждат голямо количество важни съобщения, които преминават бързо през екрана на компютъра. Често е важно да запишете тези съобщения във файл, за да можете по-късно да ги разгледате внимателно или да ги изпратите на приятел, който да ги анализира и да провери дали и къде е възникнал проблем. Затова в този раздел ще опишем накратко механизма за пренасочване на потоците с данни - една много полезна възможност на командните интерпретатори на Unix. Ако имате опит с MS-DOS, вероятно сте виждали подобна, макар и много по-ограничена реализация на същия механизъм.

Ако след произволна команда и параметрите й въведете символа > и името на файл, всички съобщения, които програмата извежда по време на изпълнението си, ще бъдат записани в този файл. Например, за да запишете резултата от изпълнението на *Is,* можете да въведете следната команда:

```
$ Is /usr/bin > -/Binaries
```

След тази команда във файла *Binaries* в личната ви директория се записва списък с файловете в директорията */usr/bin.* Ако файлът *Binaries* е съществувал, съдържанието му ще бъде изтрито и заменено с изхода на командата ***Is.*** Изтриването на съдържанието на съществуващ файл е често срещана грешка. Ако използвате команден интерпретатор *csh* или *tcsh,* можете да предотвратите това с командата:

```
$ set noolobber
```

В *bash* можете да постигнете същия ефект с командата:

```
$ noclobber=1 Не е задължително да е 1; всяка стойност има същия ефект.
```

Друг (и може би по-полезен) начин да избегнете изтриването на съдържанието на файла е да добавите новия текст в края му. В нашия пример, след като сме записали съдържанието на */usr/bin* във файла *Binaries,* можем в края му да добавим списък с файловете в */bin.* За целта трябва да използваме низа »:

```
$ Is /bin » -/Binaries
```

Техниката за пренасочване на изхода на програма е много полезна, когато често изпълнявате дадена програма и искате да запазите съобщенията, които тя извежда - нерядко срещана практика при отстраняването на проблеми.

Повечето Unix програми имат два изходни потока. Единият се нарича стандартен изходен поток, а другият - изходен поток за отпечатване на грешки. Това не е новост за програмистите на C - файлът *stderr* съответства на изходния поток за отпечатване на грешки и често се използва за отпечатване на диагностична информация.

Символът > не пренасочва изходния поток за отпечатване на грешки.
Това е полезно, когато искате да запазите нормалния изход на програмата, без да го смесвате със съобщенията за грешки. Обаче какво да правим, ако са ни необходими точно съобщенията за грешки - например, когато се опитваме да отстраним някой проблем? Едно от възможните решения е да използваме низа >& (тази конструкция работи с всички командни интерпретатори с изключение на оригиналния Bourne shell). Резултатът е пренасочване едновременно и на двата изходни потока. Например командата:

```
$ gcc invinitjig.c >S error-msg
```

ще запише всички съобщения от компилатора *gcc* във файл с име *error-msg* (разбира се, обектният код ще се запише както обикновено във файла *invinitjig.o).* В Bourne shell и *bash* бихте могли да направите същото като използвате низа &>:

```
$ gcc invinitjig.c &> error-msg
```

Сега да направим нещо наистина полезно като запишем само съобщенията за грешки, но не и нормалния изход от програмата - тоест изходния поток за отпечатване на грешки, но не и изходния поток за отпечатване на нормални съобщения. Ако използвате съвместим с Bourne shell команден интерпретатор, можете да направите това със следната команда:

```
$ gcc invinitjig.c 2> error-msg
```

Командният интерпретатор използва числото 1 за означаване на изходния поток за нормални съобщения, а числото 2 - за потока за отпечатване на грешки. Затова горната команда ще запише в *error-msg* само съобщенията за грешки.

И накрая, нека направим така, че нормалните съобщения да не се извеждат на екрана. Обикновено това се постига чрез пренасочване на стандартния изходен поток към специалния файл */dev/null* (сигурно сте чували изрази като "Кажи мнението си на /dev/null" - е, вече знаете откъде идва тази фраза). В Unix системите директорията */dev* съдържа специални файлове, които се използват за връзка с терминали, лентови устройства, твърди дискове и други устройства. Файлът */dev/null* е уникален - всички данни, които записвате в него, изчезват като в "черна дупка". Например, следващата команда записва съобщенията за грешки във файл и не извежда на екрана нормалните съобщения:

```
$ gcc invinitjig.c 2>error-msg >/dev/null
```

По този начин можете да изолирате точно съобщенията, които искате.

Ако сте се чудили дали символа < означава нещо за командния интерпретатор - да, означава. С този символ се пренасочва входния поток - така можете да накарате дадена програма да чете входните си данни от файл. Все пак, повечето програми ви позволяват да зададете входните файлове от командния ред, затова рядко се налага да използвате пренасочване на входния поток.

Понякога е полезно една програма да обработва данните, изведени от друга програма. Например, бихте могли да използвате командата *sort,* за да подредите по удобен за вас начин текста, отпечатан от друга програма. Един не особено добър начин да направите това е да запишете резултата от избраната команда в междинен файл, след което да подадете този файл като параметър на командата *sort:*

```
$ du > du_output
$ sort -n du_output
```

Unix предлага значително по-просто и ефективно решение на проблема - така наречените *канали* (pipes). Достатъчно е да поставите вертикална черта между първата и втората команда:

```
$ du | sort -n
```

Командният интерпретатор ще изпрати целия текст, изведен от програмата *du,* на програмата *sort.*

В този пример използвахме командата *du* (съкращение от "disk usage"), за да открием каква част от диска заема всеки файл в текущата директория и поддиректориите й. Обикновено изходът на *du* е неподреден:

```
$ du
10          /zoneinfo/Australia
13          /zoneinfo/US
9           /zoneinfo/Canada
4           /zoneinfo/Mexico
5           /zoneinfo/Brazil
3           /zoneinfo/Chile
```

```
20          /zoneinfo/SystemV
118         /zoneinfo
298         /ghostscript/doc
183         /ghostscript/examples
3289        /ghostscript/fonts
```

Затова решихме да подредим данните с командата *sort* с опции -и и *-г.* Опцията *-п* означава, че искаме сортировката да се извърши по *числата,* с които започва всеки ред, вместо да се използва подразбиращата се подредба по ASCII кодове на *символите.* Опцията *-г* означава, че искаме подредбата да е в обратен ред, така че първо да се изведе редът, започващ с най-голямото число. Като резултат от всичко това ще разберем кои файлове и директории заемат най-много място на диска:

```
$ du I sort -rn
34368
16005       ./emacs
16003       ./emacs/20.4
13326       ./emacs/20.4/lisp
4039        ./ghostscript
3289        ./ghostscript/fonts
```

Понеже файловете са твърде много, би било добре да използвате втори канал, за да изпратите резултата на командата *more* (често каналите се използват именно по този начин):

```
$ du I sort -rn I more
34368
16005       ./emacs
16003       ./emacs/20.4
13326       ./emacs/20.4/lisp
4039        ./ghostscript
3289        ./ghostscript/fonts
```

Какво е команда

Както вече споменахме, Unix предлага огромно количество команди и всеки може да добавя нови команди. Именно затова Unix е коренно различен от повечето операционни системи, които съдържат строго ограничено количество команди. И така, какво представляват командите на Unix и как се съхраняват? В Unix командата е просто един файл. Например, командата *Is* е двоичен файл, който се намира в директорията *bin.* Затова вместо *Is* можем да въведем пълния път към командата, познат още като *абсолютен път:*

```
$ /bin/Is
```

Това прави Unix много гъвкава и мощна операционна система. За да се добави нова полезна програма, достатъчно е системният администратор просто да я инсталира в стандартна директория, където се намират и други команди. Възможно е да се съхраняват различни версии на дадена команда - например, можете да запишете нова версия за тестване на едно място, като оставите старата версия на друго място, така че потребителите да използват версията, която предпочитат.

Ето един общ проблем: опитвате се да изпълните команда, която очаквате да се намира в системата, но получавате съобщение, че командата не е намерена ("Not found"). Причината за проблема може да е в това, че командният интерпретатор не претърсва директорията, в която се намира вашата команда. Списъкът от директории, в които командният интерпретатор търси команди, се нарича *път*. За да видите пътя за текущия потребител, въведете следната команда (не забравяйте символа $):

```
$ echo $PATH
/usr/local/bin:/usr/bin:/usr/XHR6/bin:/bin:/usr/lib/java/bin:\
/usr/games :/usr/bin/TeX:.
```

Разгледайте внимателно този текст. Той представлява последователност от имена на директории, разделени с двоеточия. В конкретния пример първата директория е */usr/local/bin,* следва */usr/bin* и т.н. Ако съществуват две версии на дадена команда и едната се намира в */usr/local/bin,* а другата - в */usr/bin,* ще се изпълни версията в */usr/local/bin*. Последното име в този пример е *точка* и съответства на текущата директория. За разлика от MS-DOS, Unix не претърсва автоматично и текущата директория - би трябвало това изрично да го зададете, както е показано тук. Някои хора смятат, че автоматичното претърсване на текущата директория е лоша идея, основно заради риска за сигурността на системата (възможно е някой злонамерен потребител да запише вредна програма в някоя от работните ви директории). Все пак, в повечето случаи това се отнася за потребителя root и обикновените потребители не трябва да се притесняват от такива проблеми.

Ако някоя команда не може да бъде намерена, трябва да откриете в коя директория се намира тя и да добавите тази директория към пътя си. Един от възможните начини да намерите в коя директория се намира дадена команда, е да използвате справочната й страница. Да предположим, че сте открили командата в директорията */usr/sbin* - в нея се намира голяма част от командите за системно администриране. Ако смятате, че ви е необходим достъп до тези команди, бихте могли да въведете следното (забележете, че в първия **PATH** няма символа $, а във втория има):

```
S export PATH=$PATH:/usr/sbin
```

Тази команда добавя директорията *Aisr/sbin* към пътя, но така, че тя да бъде последната директория, в която се търсят команди. Или, казано по друг начин - "Искам пътя ми да бъде стария път плюс */usr/sbin".*

Този пример работи с някои командни интерпретатори, но не работи с други. Той е подходящ за повечето потребители на Linux, които използват команден интерпретатор, съвместим с Bourne shell, например *bash.* Но ако използвате *csh* или *tcsh,* вместо горната трябва да изпълните следната команда:

```
set path = ( $PATH /usr/sbin )
```

И накрая, съществуват няколко команди, които не са файлове. Една от тях е *ed.* Повечето от тези команди се отнасят за самия команден интерпретатор и затова трябва да бъдат разпознати и изпълнени от него. Тези команди са част от командния интерпретатор, поради което се наричат вградени команди.

Поставяне на команда във фонов режим

Преди да бъде създадена системата X Window (с която е лесно да се използват едновременно няколко програми), потребителите на Unix използваха многозадачните му възможности просто като добавяха символа & в края на командата, както е показано в следващия пример:

```
S gec invinitjig.c &
[1] 21457
```

Символът & се използва за поставяне на командата във фонов режим, което означава, че командният интерпретатор веднага ще изведе покана за въвеждане на следваща команда, без да изчаква края на изпълнението на предишната. Това ви позволява да изпълнявате и други команди докато *gec* компилира изходния код. `[ 1 ]` е така наречения номер на задачата, който е асоцииран с вашата команда. Числото `21457` е идентификатор на процеса и ще го разгледаме по-късно. Номерата на задачите се задават последователно на работещите във фонов режим команди и затова са малки числа, които се запомнят и въвеждат по-лесно, отколкото идентификаторите на процесите.

Разбира се многозадачността не е безплатна. Колкото повече команди поставяте във фонов режим, толкова по-бавно ще работи системата, защото се опитва да отдели време за всяка задача.

Не би трябвало да поставяте във фонов режим команди, които изискват от потребителя да въвежда данни. Ако го направите, системата ще изведе съобщение за грешка, например:

```
Stopped (tty input)
```

Можете да решите проблема като зададете нормален режим (foreground mode) за изпълнение на задачата. За целта трябва да използвате командата^. Ако в момента във фонов режим се изпълняват много команди, можете да изберете една от тях с помощта на номера на задачата или идентификатора на процеса за тази команда. В горния пример (за компилатора *gcc)* можете да използвате една от следните еквивалентни команди:

```
S fg %i
$ fg 21457
```

Не забравяйте да въведете символа % преди номера на задачата; този символ се използва за разграничаване на номера на задачата от идентификатора на процеса.

За да спрете изпълнението на дадена команда, работеща във фонов режим, използвайте командата *kill:*

```
$ kill %1
```

Справочни страници

Най-полезната информация, която можете да получите, е как да направите собствено изследване. Като спазваме това предписание, ще ви покажем как да използвате вградената в Unix система за електронна документация. Тази система се нарича справочни страници (manual pages или съкратено man pages).

В действителност справочните страници не са толкова изчерпателни, колкото би трябвало да бъдат. Това е така, защото те съдържат сбита информация и предполагат значителни познания за Unix. Всяка страница е посветена на конкретна команда и рядко става ясно защо трябва да използвате тази команда. И все пак тези страници имат голямо значение. В различните Unix системи командите могат да имат малки изменения, а справочните страници са най-сигурния начин да разберете как точно работи дадена команда във вашата система. Хората, работещи по LDP (Linux Documentation Project - проект за документиране на Linux), заслужават голямо уважение за безкрайното време, което са отделили за създаването на справочни страници.

За да получите справочната страница за дадена команда (например *Is),* използвайте следната команда:

```
$ man Is
```

Справочните страници са разделени на няколко раздела в зависимост от предназначението си. Потребителските команди са в първи раздел, системните функции на Unix са във втори раздел и т.н. Най-интересните

раздели за начинаещи потребители са 1, 5 (файлови формати) и 8 (команди за системна администрация). Когато използвате електронния вариант на справочните страници, можете да задавате номера на раздела, в който смятате, че ще откриете информация за дадена команда:

```
$ man 1 Is
```

Ако използвате отпечатано на хартия ръководство, ще откриете, че разделите в него са номерирани аналогично на тези от електронния вариант. Понякога два различни раздела могат да съдържат страници с едно и също заглавие (например, *chrnod* е едновременно потребителска команда и системна функция). Затова често след името на справочната страница се дава номера на раздела - например ***Is (I)***.

Когато съществуват няколко страници за една и съща ключова дума (например една за потребителска команда с това име и една за системна функция със същото име), тогава в командния ред трябва да зададете номера на раздела. Представете си, че търсите описание на системна функция, но програмата *man* ви показва потребителската команда, защото по подразбиране търси първо в раздела с потребителски команди. В такъв случай, за да видите страницата за системната функция, трябва да зададете номера на съответния раздел.

Погледнете началото на справочната страница. Първата секция е **NAME.** В нея ще намерите кратко едноредово описание на обекта, на който е посветена страницата. Това описание ще ви бъде полезно, ако не сте съвсем сигурни какво търсите. Намерете дума, която е свързана с това, от което се интересувате, и я задайте като параметър на командата *apropos:*

```
$ apropos edit
```

Тази команда ще изведе списък с всички справочни страници, които по някакъв начин са свързани с редактирането (на английски "edit"). Алгоритъмът на *apropos* е много прост: отпечатват се всички секции **NAME,** които съдържат текста, който сте задали.

xman е приложение за X Window, с което също можете да разглеждате справочните страници. Програмата *xman* е описана в раздела "хтап - графичен интерфейс към справочните страници" в Глава 11, *Настройка на графичната среда*.

Глава 11

Както командите, така и справочните страници понякога са инсталирани на странни места във файловата система. Например, можете да инсталирате специфична за вашия сайт програма в директорията *hisr/local* и да поставите справочните й страници в директорията */usr/local/man*. Командата *шап* не търси автоматично справочни страници в тази директория, затова когато поискате справочната страница за вашата програма, ще получите съобщение за грешка "No manual entry" (няма такава спра-

вочна страница). Проблемът може да се реши като опишете всички главни директории, които съдържат справочни страници. Списъкът с тези директории се намира в променлива с име **MANPATH.** Например (трябва да зададете истинските директории, които съдържат справочните страници в системата ви):

```
$ export MANPATH=/usr/man:/usr/local/man
```

Синтаксисът е като при рдтн и е описан по-горе в тази глава. Всеки две директории се разделят с двоеточие. Ако командният ви интерпретатор е ***csli*** или ***tcsh,*** трябва да въведете командата:

```
$ setenv MANPATH /usr/man:/usr/local/man
```

Не се учудвайте, ако след прочитането на няколко справочни страници все още не сте получили отговор на въпроса си. Тези страници не са предвидени като въведение в материята, която описват. Най-добре е да намерите хубава книга за начинаещи потребители на Unix и след като свикнете със системата да се върнете към справочните страници - тогава те ще бъдат незаменим източник на информация.

Справочните страници не са единствения източник на информация за Unix системите. Често програмите от проекта GNU имат документация в така наречения Info формат (или info-страници), която можете да разглеждате с програмата *info.* Например, за да прочетете info-страницата за командата *find,* трябва да въведете:

```
info find
```

Програмата *info* има малко неудобен интерфейс, но много възможности за навигация; най-добрият начин да научите как да я използвате е да я стартирате, да натиснете Ctrl-H и да прочетете показаната документация. За щастие, съществуват няколко други програми, с които използването на info-страници е по-лесно, като за отбелязване са *tkinfo* и *kdehelp.* Тези програми използват системата X Window за графичния си интерфейс. Освен това, можете да четете info-страници с Emacs (виж раздел "Текстовият редактор Emacs" в Глава 9) или да използвате командата *pinfo,* която е част от някои дистрибуции и работи подобно на web-браузера Lynx.

Глава 16

В последно време се увеличава количеството на документацията във формат HTML. Тази документация може да се разглежда с произволен web-браузер (виж Глава 16, *World Wide Web и електронна поща).* Например, в Netscape Navigator трябва да изберете "Open Page..." в менюто "File", след което да натиснете бутона "Choose File". Ще се отвори диалогов прозорец, в който можете да изберете файла с документация.

Собственост и права за достъп до файл

Собствеността и правата за достъп до файловете са в основата на системата за сигурност в Unix. Важно е да настроите правилно тези параметри, дори ако вие сте единствения потребител на системата си, защото в противен случай могат да се получат странни неща. Тези настройки не са особено важни за файловете, които обикновените потребители създават и използват в ежедневната си работа (макар че е полезно да се знаят основните принципи). Но нещата не са толкова прости, когато става дума за системната администрация. Ако са зададени неправилно собствеността и правата за достъп до някои файлове, е възможно да не работи например системата за електронна поща. В общия случай, съобщението:

```
Permission denied
```

означава, че правата за достъп до някой файл са ограничени повече, отколкото бихте искали.

Какво означава право за достъп

Правата за достъп до даден файл определят начините, по които потребителите мотат да използват този файл. В Unix съществуват три такива права:

- Право за *четене* означава, че потребителят може да чете съдържанието на файла.
- Право за *писане* означава, че потребителят може да променя съдържанието на файла или да го изтрива.
- Право за *изпълнение* означава, че потребителят може да стартира файла като програма.

Когато създавате файл, системата автоматично задава няколко подразбиращи се права за достъп до него. В повечето случаи не е необходимо те да се променят. Обикновено вие имате права за четене и писане, а останалите потребители могат само да четат създадения от вас файл. Ако имате причина да сте параноик, можете да промените правата за достъп така, че другите потребители да нямат никакви права върху вашия файл.

Освен това, повечето програми знаят как да задават права за достъп. Например, когато компилаторът създава изпълнима програма, той автоматично задава право за изпълнение. Ако използвате система за управление на промените (revision control system или RCS) и вземете от нея даден файл, без да го заключите, имате само право за четене (защото не се очаква да променяте файла), но ако го заключите, получавате право

Глава14

за четене и право за писане (защото се очаква да редактирате файла и да го върнете, след като приключите). Ще разгледаме ЯСБ в раздела "Системата за управление на промените RCS" в Глава 14, *Инструменти за програмисти*.

В някои случаи файловете, които създавате, трябва да имат различни от подразбиращите се права за достъп. Например, ако създадете скрипт за командния интерпретатор или програма на Perl, трябва да си дадете право за изпълнение, за да можете да стартирате файла. Ще ви покажем как става това по-долу в този раздел, след като обясним основните концепции.

Когато правата за достъп се отнасят за директория, те имат друго значение:

- Право за четене означава, че потребителят може да прочете съдържанието на директорията.
- Право за писане означава, че потребителят може да добавя и изтрива файлове в директорията.
- Право за изпълнение означава, че потребителят може да получи информация за файловете в директорията.

Не се притеснявайте за разликата между правото за четене и правото за изпълнение; обикновено или едновременно ги имате, или едновременно ги нямате.

Забележете, че ако разрешите на потребителите да добавят файлове в дадена директория, едновременно с това те получават право да изтриват файлове. Тези две привилегии следват от дефиницията на право за писане в директория. Все пак, съществува начин потребителите да използват съвместно дадена директория, без да имат право да изтриват чужди файлове (виж раздел "Актуализиране на друг софтуер" в Глава 7, *Актуализиране на софтуера и ядрото)*.

Освен обикновените файлове и директории, за които говорихме досега, в Unix съществуват и други типове файлове. Това са специални файлове за комуникация с хардуерни устройства, sockets (използват се за комуникация между процеси), символни връзки и т.н. Различните типове влагат свой смисъл в правата за достъп, но в момента не е необходимо да знаете всички детайли.

Собственици и групи

И така, кой получава правата за достъп? За да могат много потребители да работят независимо един от друг на една и съща система, Unix различава три нива на достъп: собственик, група и други. "Други" са всички

потребители, които имат достъп до системата и не са собственика или член на групата, асоциирана с файла.

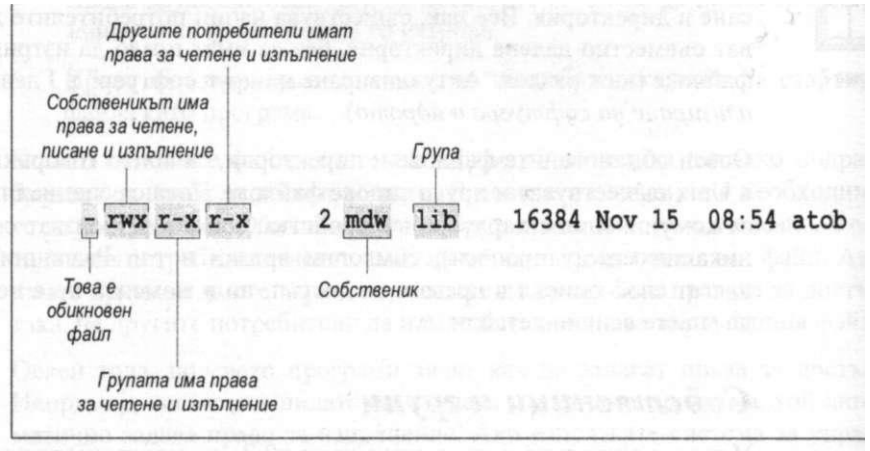
Причината за съществуването на групи е следната: често няколко потребители, например екип от програмисти, работят заедно върху определен проект и затова трябва да имат достъп до даден файл. Програмистът, който създава изходния код в този файл, запазва за себе си право да го променя, но дава на потребителите от своя екип право да четат файла чрез правата на групата за достъп до файла. Всички останали потребители извън групата могат да нямат никакви права върху този файл. (Мислите ли, че вашият код е *чак толкова* добър?).

Всеки файл има собственик и асоциирана група. Собственик обикновено е потребителят, който е създал файла. Всеки потребител принадлежи на една подразбираща се група и тази група автоматично се асоциира с всеки файл, който създава въпросният потребител. Възможно е в системата да съществуват много групи и всеки потребител да принадлежи на няколко от тях. Ако смените групата, асоциирана с даден файл, можете да дадете достъп до него на избрани от вас хора. Ще обсъдим групите по-подробно, когато стигнем до раздела "Файлът /etc/group" в Глава 5.

Глава 5

Вече имаме всички елементи в нашата система за сигурност: три права за достъп (четене, писане и изпълнение) и три нива на достъп (собственик, група и други). Да разгледаме някои типични файла и правата за достъп до тях.

На Фигура 4-2 са показани правата за достъп до една типична изпълнима програма. Този текст е генериран от командата *Is* с опция -/.



Фигура 4-2: Собственост и права за достъп

Изпъкват два интересни факта: собственик на файла е съавторът на тази книга и ваш верен водач **mdw,** а групата е **lib** (може би група, създадена за програмисти, работещи с библиотеки). Но ключовата информация за правата за достъп е кодирана с няколко букви в лявата част на екрана.

Първият символ е **тире** и означава, че това е обикновен файл. Следващите три символа се отнасят за собственика и както можем да очакваме, **mdw** притежава и трите права за достъп. Втората група от три символа се отнася за членовете на групата - те могат да четат файла (което не е особено полезно, когато работим с двоичен файл) и да го изпълняват, но не могат да пишат в него, защото полето, което би трябвало да съдържа символа **w,** вместо него съдържа **тире.** Последните три символа се отнасят за "останалите", които в този случай имат същите права, както и групата.

Като друг пример ще разгледаме файл, който е взет за редактиране от системата RCS:

```
-rw-r—r—   2 mdw      lib           878 Aug  7 19:28 tools.tex
```

Единствената разлика между този файл и показания на Фигура 4-2 е в това, че символът **x** в този случай е заменен с **тире.** Не е необходимо някой да има право за изпълнение, защото файлът не е изпълним - той е просто един текстов файл.

Още един пример - типична директория:

```
drwxr-xr-x   2 mdw      lib           512 Jul 17 18:23 perl
```

В този случай най-левият символ е d, което показва, че *lib* е директория. Символите **x** отново са тук, защото бихме искали хората да виждат съдържанието на директорията.

Справочната страница на ***Is*** съдържа изчерпателна информация за различните полета и възможните стойности в тях. Сега обаче е време да видим как могат да се променят собствеността и правата за достъп до файл.

Промяна на собственика, групата и правата за достъп до файл

Както казахме, в повечето случаи не е необходимо да променяте правата за достъп, които са зададени по подразбиране по време на създаването на файла. Обаче винаги съществуват изключения от това правило, особено за системните администратори.

Като пример ще вземем създаването на лична директория за нов потребител. За целта трябва да влезете в системата като потребителя root и

след това да извършите всичко необходимо. Обаче файловете и директориите, които създавате, ще бъдат собственост на root. Затова след като свършите, трябва да направите новия потребител собственик на създадените за него файлове и директории - в противен случай, той няма да може да ги използва! (Това се извършва автоматично от командата *adduser* - виж раздел "Създаване на нов account" в Глава **5.)**

Глава 5

По подобен начин работят някои програми като UUCP и News, които имат свои собствени account-и. Никой не може да влезе в системата като UUCP или News, но такива потребители и групи трябва да съществуват, за да могат програмите да извършат работата си като се спазват правилата за сигурност. Обикновено последната стъпка, която трябва да извършите след като сте инсталирали софтуер, е да промените собственика, групата и правата за достъп по начина, описан в документацията на софтуера.

Командата *chown* променя собственика, а командата *chgrp* променя групата, асоциирана с даден файл. В Linux само потребителят root може да използва командата *chown,* за да сменя собственика на даден файл, но всеки потребител може да зададе нова група за файл, който е негова собственост (потребителят трябва да принадлежи на новата група).

И така, ако сте инсталирали софтуер с име *sampsoft,* можете да промените собственика на файла и групата на bin, като използвате следните команди:

```
# chown bin sampsoft
# chgrp bin sampsoft
```

Можете да извършите същото на една стъпка като използвате записа:

```
# chown bin.bin sampsoft
```

Синтаксисът на командата за промяна на правата за достъп е по-сложен. Правата за достъп се наричат още "file mode" (начин за използване на файла), затова командата за промяната им се нарича *chmod.* Ще започнем изследването на тази команда с един прост пример - написали сте програма на Perl или Tel с име *header* и искате да имате право да я изпълните. Бихте могли да използвате следната команда:

```
$ chmod +x header
```

Знакът + означава "добави право за достъп", а **x** задава правото, което трябва да се добави.

Ако искате да премахнете правото за изпълнение, вместо плюс трябва да зададете минус :

```
$ chmod -x header
```

В този вид командата *chmod* задава права за достъп едновременно за всички нива - собственика, групата и останалите потребители. Да предположим обаче, че тайно създавате софтуер и не искате никой друг да може да го изпълнява (не, това не е твърде жестоко - ако в скрипта има грешка, по този начин предпазвате останалите потребители, докато не отстраните грешката). Можете да зададете право за изпълнение само за себе си с командата:

```
$ chmod u+x header
```

Символът, който се намира пред плюса, показва нивото на достъп, а символът след него - правото за достъп. Нивото на достъп за собственика е u (от user permission или потребителски права за достъп), за групата е g (от group permission), а за останалите потребители е o (от others permission). За да зададете право за достъп едновременно на вас и на групата, трябва да въведете:

```
S chmod ug+x header
```

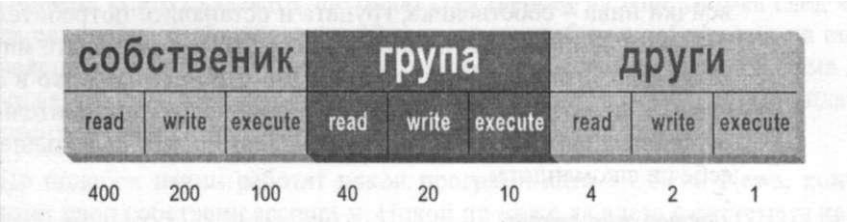
Можете да зададете едновременно няколко права за достъп:

```
$ chmod ug+rwx header
```

Съществуват още няколко такива команди, които можете да научите от справочната страница на *chmod.* С тях можете да впечатлите някой, който гледа през рамото ви какво правите, но не предлагат много по-голяма функционалност от тази, която ви показахме.

Макар че описаният по-горе синтаксис на *chmod* изглежда достатъчно сложен, съществува още един синтаксис, който е дори по-мощен. Налага се да го опишем поради няколко причини. Първо, съществуват някои ситуации, които не могат да бъдат описани с горния синтаксис, наречен *символен синтаксис.* Второ, в някои документации се използва друг синтаксис - така наречения *пълен синтаксис.* Трето, има ситуации, в които действително е по-удобно да се използва пълния синтаксис.

За да разберете пълния синтаксис, трябва да можете да работите с битове и осмична бройна система. Не се притеснявайте, това не е много трудно. Типичните права за достъп съдържат три цифри, съответстващи на трите нива на достъп (собственик, група и други). Това е показано на Фигура 4-3. Във всяко ниво има три бита, които съответстват на правата за четене, писане и изпълнение.



Фигура 4-3: Битовете в пълния синтаксис

Да предположим, че искате да дадете на себе си право за четене, а всички останали да нямат никакви права. Трябва да зададете само бита, който съответства на числото 400. Тогава командата *chmod* ще бъде:

```
$ chmod 400 header
```

За да дадете право за четене на всеки потребител, трябва да изберете съответните битове от всяко ниво - 400 за собственика, 40 за групата и 4 за останалите. Пълната команда е:

```
$ chmod 444 header
```

Действието на *chmod* е като аргумент +r, с изключение на това, че едновременно се премахват правата за писане и изпълнение. (За да бъдем съвсем точни, командата действа точно както при аргумент =r. Символът "равно" означава "правата ще са тези и само тези".)

За да дадете право за четене и право за изпълнение на всеки потребител, трябва да изчислете сумата от битовете, които отговарят на правата за четене и правата за изпълнение (например 400 + 100 е 500).

И така, съответната команда е:

```
$ chmod 555 header
```

което е същото като =rx. За да разрешите пълен достъп за някое ниво, трябва съответната му цифра да бъде 7 - сумата на 4, 2 и 1.

Една последна особеност: как се задават правата за достъп, които се използват по подразбиране при създаване на файл (например с текстов редактор, при пренасочване на изходния поток с > и т.н.)? Това се прави с командата *umask*. Ако искате тези права да се задават автоматично всеки път, когато влизате в системата, трябва да добавите *umask* във файла, който се изпълнява автоматично при стартиране на командния интерпретатор. Това може да е *.bashrc, .cshrc* или някой друг файл, в зависимост от командния интерпретатор, който използвате (ще разгледаме тези конфигурационни файлове в следващия раздел).

Командата ***iimask*** има един параметър, в който се използва пълния синтаксис на ***chmod,*** но значението на битовете е обратното. След като изберете правата за достъп на собственика, групата и останалите, трябва да замените всяка цифра с резултата от израза (7 - съответната цифра). Така ще получите маска от три цифри.

Да предположим, че искате вие да имате всички права за достъп (7), вашата група да има права за четене и изпълнение (5), а останалите да нямат никакви права (0). След като от 7 извадите всяка цифра, ще получите съответно 0 за вас, 2 за групата ви и 7 за останалите. Командата, която трябва да изпълните е:

```
umask 027
```

Странна техника, но работи. Командата *chmod* използва тази маска, когато интерпретира правата за достъп. В дадения пример файловете ще се създават с право за изпълнение, като *chmod* ще даде това право за вас и групата ви, но няма да го даде на останалите потребители, защото избраната маска не им разрешава никакъв достъп.

Конфигурационни файлове

Конфигурирането е една от силните страни на Unix. Вероятно причината за това са две отличителни черти на хакерите - те искат пълен контрол върху работната си среда и се стремят да намалят броят на клавишите, които трябва да натиснат. Затова всички важни приложения за Unix - редактори, програми за работа с електронна поща и за изчистване на грешки, клиенти за системата X Window - използват конфигурационни файлове, с които можете да промените поведението им по невероятно количество начини. Много от тези файлове имат имена, завършващи на *гс* - съкратено от *resource configuration* (конфигурация на ресурсите).

Конфигурационните файлове (наричат се още startup файлове - файлове, които се изпълняват при стартиране) обикновено се намират в личната ви директория. Техните имена започват с точка и затова командата ***Is*** не ги отпечатва, ако не е зададена специална опция за това. Нито един от тези файлове не е задължителен; всяка програма, която използва конфигурационен файл, работи с подразбиращите се параметри, ако този файл липсва. Почти всички потребители смятат, че е полезно да използват конфигурационните файлове. Ето някои от най-важните:

.bashrc

Използва се от командния интерпретатор *bash.* Файлът представлява скрипт на командния интерпретатор, което означава, че може да съдържа команди и други програмни конструкции. Следва пример

за кратък конфигурационен файл, който може да бъде поставен в личната ви директория от програмата, с която е създаден вашият account:

```
# Поканата за въвеждане на команда съдържа името на потребителя.
PS1='\u $'

# Запомнят се последните 50 въведени команди.
HISTSIZE=50

# Директориите, в които командният интерпретатор
# ще търси команди.
PATH=/usr/local/bin:/usr/bin:/bin:/usr/bin/X11

# За да предпазим потребителя от неволно прекратяване
# на сесията му, забраняваме клавишната комбинация
# Ctrl-D като команда за изход.
IGNOREEOF=1

# За да сме сигурни, че клавишът Backspace работи коректно.
stty erase ^H
```

.bash _profile

Използва се от командния интерпретатор ***bash.*** Разликата между този скрипт и ***.bashrc*** е в това, че ***.bashjprofile*** се изпълнява само когато влизате в системата. Първоначалната цел е била да се разделят интерактивните командни интерпретатори от тези, които се пускат от фонови процеси, например от cron (виж Глава **8,** *Други административни задачи).* Това не е особено полезно в модерните компютри, които използват системата X Window, защото когато отворите нов прозорец на програмата ***xterm,*** се изпълнява само ***.bashrc.*** Ако отворите прозорец с командата ***xterm -Is,*** ще се изпълни и скрипта ***.bash _profile.***

.cshrc

Използва се от командния интерпретатор C shell или ***tcsh.*** Файлът съдържа shell-скрипт с програмни конструкции на C shell.

.login

Използва се от командния интерпретатор C shell или ***tcsh.*** Съдържа shell-скрипт с програмни конструкции на C shell. Този скрипт се изпълнява само когато влизате в системата (аналогично на ***.bash_profile*** при командния интерпретатор bash). Ето няколко команди, които можете да намерите в ***.cshrc*** или ***.login:***

```
# Символът % ще се използва като покана за въвеждане на команди.
set prompt='% '

# Запомнят се последните 50 въведени команди.
set history=50
```

```
            # Директориите, в които командният интерпретатор
            # ще търси команди.
            set PATH=(/usr/local/bin /usr/bin /bin /usr/bin/X11)
# За да предпазим потребителя от неволно прекратяване
            # на сесията му, забраняваме клавишната комбинация Ctrl-D
            # като команда за изход.
            set ignoreeof
# За да сме сигурни, че клавишът Backspace работи коректно.
            stty erase ^H
```

.emacs

Използва се от текстовия редактор Emacs. Съдържа функции на езика LISP (виж раздела "Настройки на Emacs" в Глава 9).

Глава 9

.exrc

Използва се от текстовия редактор *vi* (познат също като *ex).* Всеки ред съдържа една команда на редактора (виж раздела "Възможности за надстройка на vi" в Глава 9).

.fvwm2rc

Използва се от window manager-а *fvwnü.* Състои се от специални команди, които се интерпретират от *fvwm2.* Можете да намерите един такъв примерен файл в раздела "Конфигуриране на fvwm" в Глава 11, *Настройка на графичната среда.*

Глава 11

.twmrc

Използва се от програмата за управление на прозорци twm. Състои се от специални команди, които се интерпретират от *twm.*

.newsrc

Използва се от програмите за четене на групите в Usenet. Съдържа списък с групите, за които е абониран потребителят.

.Xdefaults

Използва се от програмите, работещи под системата X Window. Всеки ред съдържа името на ресурс (обикновено името на програма и определено нейно свойство) и стойността на ресурса. Този файл е описан в раздела "База данни с ресурси за X клиенти" в Глава 11.

.xinitrc

Използва се от системата X Window. Съдържа shell-скрипт, който се изпълнява когато стартирате X (виж раздела "Основни принципи при настройката на X" в Глава 11 за повече информация).

По-важни директории

Вече ви запознахме с директорията */home,* в която се съхраняват файловете на потребителите. Като системен администратор и програмист, за вас са важни и няколко други директории. Ето част от тях, заедно със съдържанието им:

/bin

Най-важните команди на Unix, например *Is.*

/nsr/bin

Други команди. Разпределението на файловете между */bin* и */usr/bin* е до голяма степен случайно; причината за съществуването на две директории *bin* е, че ранните Unix системи са имали малки по обем дискове и по този начин е било лесно командите да се разпределят между тях.

/usr/sbin

Командите, които се използват за системна администрация.

/boot

Обикновено тук се записват ядрото и други файлове, които се използват при зареждането на системата.

/etc

Конфигурационни файлове за различни подсистеми, например за мрежовата подсистема, NFS, електронна поща и др. Обикновено тези файлове съдържат таблици с мрежовите услуги, дисковете, които трябва да се монтират и т.н.

/var

Административни файлове (например log-файлове), които се използват от различни програми.

/var/spool

Място, в което временно се съхраняват файловете, които се отпечатват в момента или се изпращат от UUCP и т.н.

/usr/lib

Стандартни библиотеки като *libc.a.* Когато компилирате някоя програма, свързващият редактор винаги търси в тази директория библиотеките, зададени с опцията -/.

hisr/lib/Xll

Дистрибуцията на системата X Window. Съдържа библиотеките, които се използват от X клиентите, както и шрифтове, примерни кон-

фигурационни файлове и други важни части от пакета X. Обикновено тази директория е символна връзка към */usr/Xl lR6/lib/Xl 1.*

/usr/include

Стандартното място за заглавните файлове за програмите на C, например за заглавния файл *stdio.h.*

/usr/src

Тук се съхраняват изходните текстове за готовите програми, които компилирате на системата си.

/usr/local

Програми и файлове с данни, които са добавени локално от системния администратор.

/etc/skel

Примерни конфигурационни файлове, които се поставят в личните директории на новите потребители.

Обслужващи програми

Добавяме този раздел, защото предполагаме, че вече ви е интересно какви програми работят паралелно с вас в системата.

Много от дейностите в една модерна компютърна система са твърде сложни. Понякога за извършването им не е достатъчно просто да се прочете даден файл или друг статичен ресурс - необходимо е взаимодействие с друг работещ процес.

Като пример ще вземем протоколът FTP, който вероятно сте използвали, за да изтеглите от Интернет софтуер или документи, свързани с Linux. Когато използвате FTP, за да се свържете с друга система, на нея трябва да работи програма, която да осъществи връзката с вашия компютър и да обслужи заявките ви. Тази програма се нарича *ftpd.* Буквата *d* идва от *демон* (daemon) - терминът, който се използва в Unix, за да се обозначи сървър, който работи през цялото време като фонов процес в системата. Повечето демони обслужват мрежови дейности.

Вероятно ви е омръзнало да слушате мантрата *клиент/сървър,* но в Unix тази система действително работи, и то от години.

Демоните се стартират при зареждането на системата. За да видите как става това, разгледайте файловете */etc/inittab* и */etc/inetd.conf,* както и специфичните за вашата дистрибуция конфигурационни файлове. Не бихме искали тук да обсъждаме техния формат. Всеки ред от тези файлове показва програма, която се стартира при зареждане на системата. Можете да разберете кои са файловете, специфични за вашата дистрибу-

ция, като проверите документацията й или като видите кои пътища се срещат най-често в */etc/inittab* -обикновено те показват коя директория се използва за съхранение на специфичните за вашата дистрибуция конфигурационни файлове.

Като пример за това как вашата система използва */etc/inittab* можете да разгледате редовете, които съдържат низовете getty или agetty. Това е програмата, която следи терминала (tty), и чака потребителя да влезе в системата. Същата програма отпечатва поканата за влизане в системата login:, за която говорихме в началото на тази глава.

Файлът */etc/inetd.conf* е пример за по-сложен алгоритъм за стартиране на програми и представлява още едно ниво на абстракция. Идеята зад */etc/inetd.conf* е следната - биха се загубили много системни ресурси, ако във всеки момент работят на празен ход дузина или повече демони и очакват по мрежата да дойде заявка, която да обслужат. Затова вместо тях системата стартира един единствен демон с име *inetd,* който очаква заявка от друг компютър и когато тя пристигне, стартира съответния демон, който трябва да я обработи. Например, когато пристигне заявка за FTP връзка, *inetd* стартира FTP демона *(ftpd),* който ще обслужи тази заявка. По този начин в системата работят само тези мрежови демони, които действително се използват.

В следващия раздел ще ви покажем как можете да разберете кои демони работят в системата ви. Съществува демон за всяка услуга, която вашата система предлага на другите машини в мрежата - *fingerd* обслужва у?и£е/--заявките, *rwhod* обслужва nWio-заявките и т.н. Няколко демона извършват дейности, които не са свързани с мрежата, например *kerneld,* който се грижи за автоматичното зареждане на модули на ядрото.

Процеси

В сърцевината на Unix е понятието процес. Трябва да го разбирате, за да можете да използвате ефективно системата, дори ако сте обикновен потребител; ако сте системен администратор, това понятие е още по-важно.

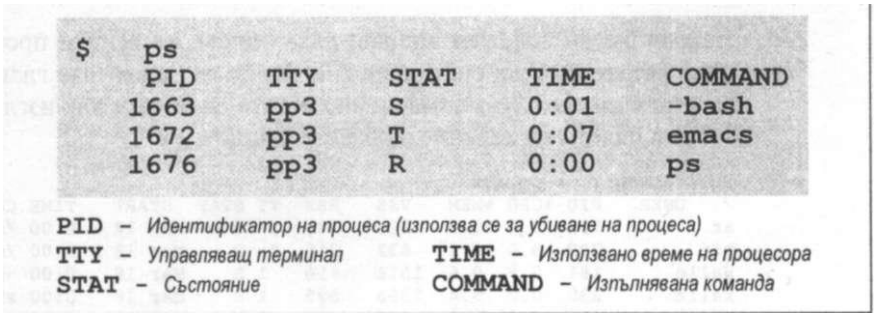
Процесът представлява самостоятелно работеща програма със свое собствено множество ресурси. Например, в един от предишните раздели ви показахме как можете да пренасочите изхода на програма във файл, като едновременно с това командният интерпретатор продължава да отпечатва съобщения на екрана ви. Причината командният интерпретатор и другата програма да могат да печатат на различни места, е че са отделни процеси.

В Unix всички ограничени ресурси на системата като памет и дискове, се управляват от една всесилна програма, наречена ядро. Всичко останало в системата са процеси.

Например, преди да влезете в системата, терминалът ви се наблюдава от процеса ***getty.*** След като влезете в системата, процесът ***getty*** "умира" и терминалът ви започва да се управлява от командния интерпретатор, който е друг процес. (Когато излезете от системата се стартира нов процес ***getty.).*** Всеки път, когато въвеждате команда, командният интерпретатор създава нов процес. Създаването на нов процес се нарича *разклоняване,* защото един процес се "разделя" на два процеса.

Ако използвате системата X Window, всеки процес стартира един или повече прозорци. Например прозореца, в който пишете команди, е собственост на процеса xterm. Този процес създава нов процес за командния интерпретатор, който работи в прозореца. А командният интерпретатор от своя страна създава още процеси за командите, които въвеждате.

За да видите кои процеси работят в компютъра ви, въведете командата *ps.* На Фигура 4-4 е показан типичен резултат от изпълнението на команда *ps* и значението на отделните полета. Сигурно ще се учудите като разберете колко много процеси използвате, особено ако в момента работи X. Един от процесите е самата команда *ps,* която разбира се умира веднага след като отпечата показаните данни.



Фигура 4-4: Резултат от изпълнението на командата ps

Първото поле във Фигура 4-4 е уникален идентификатор на процеса. Ако в машината ви работи излишен процес, който не можете да премахнете с клавишната комбинация Ctrl-C или по-друг начин, можете да го убиете като в друга виртуална конзола или в нов прозорец под X въведете командата:

```
$ kill идентификатор-на-процеса
```

Полето **TTY** показва номера на терминала, на който работи процесът, ако той използва терминал. (Всеки процес, пуснат от командния интерпретатор, използва терминал, но демоните, които работят във фонов режим, нямат терминал).

Полето **STAT** показва в какво състояние е процесът. В момента работата на командния интерпретатор временно е преустановена, затова състоянието му е **S** (от suspend). Сесията на текстовия редактор Emacs работи, но активната й дейност е прекратена с Ctrl-Z, затова в неговото поле **STAT** пише т. На последно място е процесът *ps,* който генерира тези данни, и разбира се, работи, затова в полето му пише **R** (от running).

Полето **TIME** показва каква част от времето на централния процесор са използвали показаните процеси. Понеже *bash* и Emacs са интерактивни, те всъщност не използват много от времето на процесора.

Възможно е да видите не само собствените си процеси. Отделете една минута, за да разгледате всички процеси в системата си. Ако зададете опцията *a* (идва от all, тоест "всички"), ще видите всички процеси, а опцията ***x*** включва и процесите, които нямат управляващ терминал (например демоните, които се стартират автоматично при зареждане на системата):

```
$ ps ax | more
```

По този начин можете да видите демоните, които споменахме в предишния раздел.

Накрая ще ви покажем спиращ дъха списък на всички процеси в една цяла работеща Unix система, с което ще завършим тази глава (редовете са отрязани след 76-я символ; ако искате да видите как изглежда резултатът в цялото му величие, добавете опцията w):

```
kalle@tigger: > ps aux
    USER   PID %CPU %MEM   VSZ   RSS  TT STAT  START   TIME COMMAND
at         724  0.0  0.2   824   348  ?  S    Mar 18   0:00 /usr/sbin
bin        703  0.0  0.2   832   316  ?  S    Mar 18   0:00 /usr/sbin
kalle      181  0.0  0.6  1512   856   1 S    Mar 18   0:00 -bash
kalle      230  0.0  0.4  1396   596   1 S    Mar 18   0:00 sh /usr/X1
kalle      231  0.0  0.1   808   256   1 S    Mar 18   0:00 tee /home
kalle      234  0.0  0.4  1952   624   1 S    Mar 18   0:00 xinit /hom
kalle      238  0.0  0.4  1396   616   1 S    Mar 18   0:00 sh /home/k
kalle      242  0.0  3.8  6744  4876   1 S    Mar 18   0:43 kwm
kalle      246  0.0  3.3  6552  4272   1 S    Mar 18   4:48 /usr/local
kalle      255  0.0  0.0     0     0   1 Z    Mar 18   0:00 kaudioserv
kalle      256  0.0  3.0  6208  3844   1 S    Mar 18   0:02 kwmsound
kalle      257  0.0  5.1  8892  6596   1 S    Mar 18   0:11 kfm
kalle      258  0.0  3.3  6292  4320   1 S    Mar 18   0:02 krootwm
kalle      259  0.0  4.6  7848  5988   1 S    Mar 18   0:37 kpanel
kalle      260  0.0  3.6  6764  4688   1 S    Mar 18   0:06 kbgndwm
kalle      273  0.0  3.6  6732  4668   1 S    Mar 18   0:08 kvt -resto
kalle      274  0.0  3.6  6732  4668   1 S    Mar 18   0:11 kvt -resto
kalle      276  0.0  0.6  1536   892  p0 S    Mar 18   0:00 bash
kalle      277  0.0  0.6  1512   864  p1 S    Mar 18   0:00 bash
```

```
kalle    11752  0.1  9.8 14056 12604    1 S    Mar 18   3:35 xemacs
kalle    18738  0.2 16.4 26164 21088    1 S     01:14   1:03 netscape
kalle    18739  0.0  2.6 14816  3392    1 S     01:14   0:00 (dns helpe
kalle    29744  0.0  0.3   904   428   p0 R     09:24   0:00 ps -auxw
root         1  0.0  0.2   820   292    ? S    Mar 18   0:06 init [2]
root         2  0.0  0.0     0     0    ? SW   Mar 18   0:00 kflushd
root         3  0.0  0.0     0     0    ? SW<  Mar 18   0:00 kswapd
root         8  0.0  0.2   804   264    ? S    Mar 18   0:02 update (bd
root        55  0.0  0.2   816   328    ? S    Mar 18   0:00 /sbin/kern
root        78  0.0  0.0     0     0    ? S    Mar 18   0:00 request-ro
root        96  0.0  0.3   832   408    ? S    Mar 18   0:00 /usr/sbin/
root        98  0.0  0.3   932   448    ? S    Mar 18   0:00 /usr/sbin/
root       167  0.0  0.2   824   288    ? S    Mar 18   0:00 /usr/bin/g
root       182  0.0  0.6  1508   856    2 S    Mar 18   0:00 -bash
root       183  0.0  0.2   808   288    3 S    Mar 18   0:00 /sbin/ming
root       184  0.0  0.2   808   284    4 S    Mar 18   0:00 /sbin/ming
root       185  0.0  0.2   808   284    5 S    Mar 18   0:00 /sbin/ming
root       186  0.0  0.2   808   284    6 S    Mar 18   0:00 /sbin/ming
root       235  0.3 11.8 25292 15196    ? S    Mar 18  19:19 /usr/X11R6
root       682  0.0  0.4  1076   556    ? S    Mar 18   0:00 /usr/sbin/
root       684  0.0  0.3   948   484    ? S    Mar 18   0:00 /usr/sbin/
root       707  0.0  0.3   860   440    ? S    Mar 18   0:00 /usr/sbin/
root       709  0.0  0.3   896   452    ? S    Mar 18   0:00 /usr/sbin/
root       712  0.0  0.5  1212   668    ? S    Mar 18   0:00 /usr/sbin/
root       727  0.0  0.2   840   356    ? S    Mar 18   0:00 /usr/sbin/
root       733  0.0  0.2   820   304    ? S    Mar 18   0:00 /usr/sbin/
root       737  0.0  0.2   836   316    ? S    Mar 18   0:00 /usr/sbin/
root       745  0.0  0.5  1204   708    ? S    Mar 18   0:00 sendmail:
root       752  0.0  0.4  1772   592    ? S    Mar 18   0:00 /opt/appli
wwwrun     718  0.0  0.5  1212   668    ? S    Mar 18   0:00 /usr/sbin/
wwwrun     718  0.0  0.5  1212   652    ? S    Mar 18   0:00 /usr/sbin/
wwwrun     718  0.0  0.5  1212   644    ? S    Mar 18   0:00 /usr/sbin/
wwwrun     718  0.0  0.5  1212   644    ? S    Mar 18   0:00 /usr/sbin/
wwwrun     718  0.0  0.5  1212   644    ? S    Mar 18   0:00 /usr/sbin/
```

ГЛАВА 5

Основи НАСИСТЕМНАТА АДМИНИСТРАЦИЯ

ко използвате ваша собствена Linux система, една от първите ви задачи е да се ориентирате в администрирането на системата. **X jL** Едва ли ще можете да работите дълго време без да извършите поне минимална поддръжка на системата, да актуализирате софтуера в нея или просто да направите няколко дребни промени, които са необходими, за да бъде всичко наред.

Администрирането на Linux не се различава много от карането и поддръжката на мотоциклет. Много хора предпочитат сами да се грижат за мотоциклетите си - периодично ги почистват, подменят износените части и т.н. Linux ви дава възможността да извършите същата практическа поддръжка на сложна операционна система.

Макар че един страстен системен администратор може да отдели огромно количество време за прецизна настройка на системата си, администрирането на Linux е действително необходимо, само когато трябва да се извършат значими промени в системата - инсталиране на нов диск, добавяне на нов потребител или при внезапно спиране на захранването. Тези ситуации са разгледани в следващите четири глави.

Linux е изключително гъвкава операционна система във всичките си аспекти - от рутинните задачи като актуализацията на стандартните биб-

* Някои от авторите смятат, че има тясна връзка между системната администрация на Linux и книгата на Робърт Пърсиг *"Дзен и изкуството да се поддържа мотоциклет"*. Може би Linux има същността на Буда?

лиотеки до езотеричните си възможности като настройката на ядрото.
Понеже целият изходен код е достъпен за желаещите и голяма част от разработчиците и потребителите на Linux са заклети хакери, поддръжката на системата не е просто част от ежедневния живот, а и отличен начин за придобиване на опит. Повярвайте ни, едва ли има нещо повълнуващо от това да разкажете на приятелите си как за по-малко от половин час сте преминали от X11R6.3 на X11R6.4, като едновременно с това сте прекомпилирали ядрото, за да включите поддръжка за файловата система ISO 9660 (ако те не разбират за какво говорите, можете да им дадете копие от тази книга).

В следващите глави ще изследваме Linux от техническа гледна точка - ще ви покажем какво има под повърхността и как да се грижите за системата си, включително как да актуализирате софтуера; как да управлявате потребителите, файловите системи и другите ресурси; как да архивирате данните си и какво да правите, ако възникнат проблеми. Ако досега не сте използвали Unix, ще научите и основните начини за използване и поддръжка на системата.

В повечето случаи след като настроите конфигурационните си файлове Linux ще работи дълго време, без да е необходима периодична поддръжка. Ако харесвате системната конфигурация и софтуера, който използвате, рядко се налага да администрирате системата. Все пак, горещо препоръчваме на потребителите на Linux да експериментират със системите си и да ги настройват по вкуса си. Много малко неща в Linux не могат да се променят. Ако не ви харесва как работи даден софтуер, можете просто да го промените по желания от вас начин. Например, ако предпочитате да четете мигащ зелен текст на син фон вместо традиционното черно/бяло, ще ви покажем как става това (ако обещаете, че няма да казвате откъде сте го научили).

Трябва да отбележим, че много дистрибуции включват програми, с които можете лесно да извършите голяма част от администрирането на системата. Примери за това са YaST в SuSE, LISA в Open Linux и *control-panel* или *linuxconf* в Red Hat. Тези програми могат да направят всичко от управлението на потребителски account-и до създаването на файлови системи. В зависимост от това как гледате на тях, те могат да улеснят или затруднят живота ви.

В следващите няколко глави ще ви покажем какво представлява в същината си системната администрация, като използваме само програми, които се съдържат във всяка дистрибуция на Linux и всъщност в почти всяка Unix система. Това са основните инструменти в "куфарчето" на системния администратор - метафоричните чук, отверка и гаечен ключ, на които можете да разчитате, за да си свършите работата. Ако предпочитате да използвате моторен трион - използвайте го, но винаги е по-

лезно да знаете как се работи със стандартните инструменти, в случай че стане нещо непредвидено. Ако искате да научите повече за системната администрация на Unix, препоръчваме ви да прочетете някоя от книгите *System Administration Handbook* и *Essential System Administration*.

[56] Unix !^Essential Sys.Admin

Администриране на системата

Всеки администратор на Unix система трябва да е внимателен и отговорен човек. Това важи и за Linux, дори ако вие сте единствения човек, който използва системата.

Системният администратор извършва голяма част от дейността си след като влезе в системата като потребителя **root.** Този account има специални свойства в Unix - най-вече, защото правата за достъп и други механизми за сигурност на системата просто не се отнасят за **root.** В Unix **root** има достъп и може да променя произволен файл, независимо кой е собственика му. Обикновените потребители не могат да повредят системата (например като унищожат файлова система или файлове на други потребители), но за **root** няма такива ограничения.

Защо в Unix сигурността се поставя на първо място? Една от най-важните причини за това е да се позволи на потребителите да избират начина за достъп до собствените си файлове. Всеки потребител може да промени правата за достъп до файловете си (с командата *chmod),* така че само определени групи потребители да могат да ги използват. Правата за достъп гарантират неприкосновеността и целостта на данните ви - едва ли бихте искали други потребители да четат личната ви поща или да редактират изходния текст на важна програма зад гърба ви.

Механизмите за сигурност в Unix имат и друга роля - те не позволяват на потребителите да повредят системата. Достъпът до повечето от хардуерните устройства е ограничен (виж раздела "Файлове на хардуерните устройства" в Глава 6). Ако обикновените потребители могат да четат или пишат директно в твърдия диск, вероятно ще настъпи пълен хаос - например, лесно може да бъде изтрито съдържанието на диска. Затова обикновените потребители могат да използват твърдите дискове само през файлова система, където системата за сигурност гарантира целостта на файловете с описаните в предишната глава права за достъп.

Глава 6

Важно е да се отбележи, че повредите могат да се предизвикват не само от злонамереност. Системата за сигурност е по-скоро начин да се защитят потребителите от собствените им грешки или неразбиране на определени принципи, а не опит за въвеждане на военен режим. Всъщност, в много платформи системите за сигурност са сравнително слаби. Системата за сигурност в Unix е проектирана така, че да улесни съвместната работа на групи потребители, например в разработването на определен

проект. Системата дава възможност потребителите да бъдат обединявани в групи и правата за достъп да бъдат задавани за цялата група. Например, екипът, който разработва даден проект, може да има права за четене и писане на група файлове, докато в същото време други потребители не могат да променят тези файлове. Всеки потребител може сам да реши какви да са правата за достъп до файловете му.

Освен това, механизмът за сигурност на Unix не позволява на обикновените потребители да извършват определени действия , например да използват някои системни функции в програмите си. Например, съществува системна функция, която се използва за спиране на системата и е необходима на програми като ***shutdown*** за рестартиране на компютъра (подробности за това ще намерите малко по-долу). Ако на обикновените потребители се разреши да използват тази функция в програмите си, те могат случайно (или съзнателно) да спрат системата по всяко време.

В много случаи се налага да прескочите механизмите за сигурност, за да извършите някои действия по поддръжката на системата или актуализирането на софтуера. Точно затова се използва account-а root. Понеже ограниченията не се отнасят за **root,** един опитен системен администратор може лесно да си свърши работата, без да се грижи за обичайните права за достъп или други ограничения. Обикновено за влизане като **root** се използва командата ***su.*** Тя ви позволява да използвате идентификатора на произволен потребител - например командата:

```
su andy
```

ще ви попита за паролата на **andy,** и ако я въведете, ще продължите да работите като потребителя **andy.** Системният администратор често използва временно account на обикновен потребител, за да разреши проблем с файловете на този потребител или поради друга подобна причина. Ако не въведете аргумент, програмата ***su*** ще ви попита за паролата на **root,** и след като я въведете, ще смени потребителския ви идентификатор с този на root (тоест, ще работите като **root).** След като завършите работата си като **root,** трябва да излезете от системата по стандартния начин и да се върнете към собствената си тленна същност.

Защо просто да не влезем в системата като **root** по стандартния начин? Понякога това действително е по-удобно, но в повечето случаи е по-добре да използвате ***su,*** след като сте влезли в системата със собствения си account. Ако системата ви има много потребители, командата ***su*** ще запише съобщение, подобно на това:

```
Nov  1  19:28:50  looraer  su:  mdw on /dev/ttypl
```

в някой от дневниците на системата (наричат се още log-файлове; по-долу ще опишем подробно как се използват), например във файла ***/varflog/messages.*** В конкретния случай съобщението означава, че потре-

бителят **mdw** е изпълнил успешно командата *su* и е получил идентификатора на **root.** Ако някой влезе в системата директно като **root,** това няма да се отбележи никъде в системните log-файлове и не бихте могли да разберете кой потребител е използвал този account. Това е съществено, когато системата има няколко администратора - често е важно да се знае кой и кога е използвал командата ***su.***

Account-ът **root** може да се смята за магическа пръчка - едновременно полезен и опасен инструмент. Ако си играете с вълшебните думи докато държите тази пръчка, можете да предизвикате неописуеми вреди на системата си. Като пример ще вземем простата последователност от осем символа **`rm -rf /`**, която ще изтрие всички файлове в системата ви, ако я изпълните като **root.** Мислите ли, че това е нереален пример? Съвсем не е така. Ако се опитате да изтриете стара директория, да кажем */iisr/src/oldp,* и неволно въведете интервал след първата наклонена черта, ще се получи следното:

```
rm -rf / usr/src/oldp
```

Друга често срещана грешка е объркването на параметрите на команди като *dd.* Тази команда се използва за прехвърлянето на големи блокове данни от едно място на друго. Например, ако искате да запишете във файл първите 1024 байта от устройството */dev/hda* (това включва записът за начално зареждане и таблицата с дяловете на първия твърд диск), можете да използвате командата:

```
dd if=/dev/hda of=/tmp/stuff bs=lk count=l
```

Ако обаче смените местата на **`if`** и **`of,`** ще се получи нещо съвсем различно - съдържанието на файла */tmp/stuff* ще се запише в началото на */dev/hda.* Ако */tmp/stuff* съдържа случайни или неверни данни, значи току що изтрихте таблицата с дяловете на диска си и вероятно супер-блока на файловата система. Честито, и добре дошли в чудния свят на системната администрация!

Смисълът в тези примери е следният: помислете три пъти, преди да изпълните някоя команда като **root.** Огледайте внимателно какво сте написали и се уверете, че то е правилно, преди да натиснете Enter. Ако не сте сигурни в аргументите или синтаксиса на командата, погледнете справочната й страница или се опитайте първо да я изпълните като обикновен потребител, за да установите как действа. В противен случай ще научите тези правила по трудния начин - грешките, които правите като **root,** могат да бъдат катастрофални.

В много случаи поканата за въвеждане на команди за потребителя **root** се различава от поканата за обикновени потребители. В класическия случай поканата за **root** съдържа символа #, а поканата за обикновените потребители съдържа $ или % (разбира се, само от вас зависи дали ще

използвате тази конвенция; все пак, тя се използва в много Unix системи). Макар че поканата за въвеждане на команда може да ви напомни, че размахвате магическата пръчка, нерядко потребителите забравят това или случайно въвеждат команда в погрешния прозорец или виртуална конзола.

Като с всеки мощен инструмент, с account-а **root** би могло да се злоупотреби. За всеки системен администратор е важно да защити паролата на **root,** а ако се наложи да я каже на някого, това трябва да бъде потребител, на когото може да се разчита (или човек, който може да бъде държан отговорен за действията си в системата). Разбира се, това не се отнася за система, на която вие сте единствения потребител - освен ако системата ви не е свързана в мрежа или позволява друг вид достъп.

Запазването в тайна на паролата на **root** е полезно не само за намаляването на опасността от умишлена злоупотреба, макар че това определено е много важно. Още по-важно е, че ако вие сте единствения потребител, който може да използва този account, ще знаете точно как е конфигурирана системата във всеки момент. Ако някой друг може да променя важни системни файлове (ще опишем как става това в тази глава), конфигурацията на системата може да се промени зад гърба ви и предположенията ви за това, какво точно става в компютъра, ще бъдат погрешни. Ако има само един системен администратор, той винаги знае точно как е конфигурирана системата.

Освен това, колкото повече хора знаят паролата на **root,** толкова по-голяма е вероятността за допускане на грешка при използването на този account. Дори да се доверявате на всеки човек, който знае паролата на **root,** човешко е да се допусне грешка. Ако вие сте единствения системен администратор, трябва да обвинявате само себе си за неизбежните грешки, които сте направили като **root.**

След като ви предупредихме за опасностите, ще ви покажем как да направите нещо истинско като системен администратор на Linux. Затегнете коланите.

Зареждане на системата

Съществуват много начини да заредите Linux. Най-често се използват два начина на зареждане - от дискета или от твърд диск. В повечето случаи инсталационната процедура ще конфигурира системата ви така, че Linux ще може да се зарежда по единия или и по двата начина. При всички случаи е полезно да знаете как да конфигурирате компютъра си, така че да може да зарежда Linux.

Как да използваме дискета за зареждане на Linux

Обикновено дискетата за зареждане на Linux съдържа само ядрото на операционната система, което ще се стартира при зареждане от дискета. Обикновено ядрото е компресирано с алгоритъма, който се използва в програмата *gzip* (виж раздел "Създаване на ново ядро" в Глава *1, Актуализиране на софтуера и ядрото).* Компресията се използва, за да може ядрото, което обикновено е с размер повече от мегабайт, да се събере на една дискета. Част от ядрото не е компресирана и съдържа програмния код, който трябва да зареди и разкомпресира останалата част. Затова не е необходима допълнителна програма да зарежда ядрото - то само зарежда себе си.

Глава7

В ядрото са записани няколко важни параметъра. Един от тях е устройството, което ще се използва като основна файлова система, след като ядрото приключи зареждането си. Друг параметър е текстовия режим, който трябва да се използва за системната конзола. Тези параметри могат да се променят с командата *rdev,* която ще разгледаме по-долу в този раздел.

След като приключи зареждането си, ядрото се опитва да монтира файловата система, която е кодирана в него. Това ще бъде основната файлова система, тоест /. Файловите системи са описани детайлно в раздела "Управление на файловите системи" в Глава 6; засега е достатъчно да знаете, че ядрото трябва да съдържа името на основната файлова система. Ако не успее да монтира тази файлова система, ядрото спира работата си и отпечатва съобщение за това - т.нар. kernel "panic" message, тоест "паника" на ядрото. (Всъщност, *kernel panic* е фатална грешка, която се генерира от самото ядро. Този проблем може да възникне, когато се случи нещо непредвидено и ядрото не може да продължи да работи - например, ако в самото ядро има грешка, която води до обръщение към несъществуваща памет. Ще разгледаме проблема по-подробно в раздела "Какво да правим в аварийна ситуация" в Глава 8, *Други административни задачи.)*

Главаб

Глава 8

В ядрото е кодирано името на този дял от твърдия диск, в който се намира основната файлова система. След като приключи зареждането на ядрото, този дял се монтира и управлението се прехвърля върху твърдия диск. След като ядрото се зареди в паметта, то остава там и дискетата за зареждане повече не е необходима (разбира се, докато не рестартирате системата си).

В някои дискети е инсталирана програмата LILO, която може да зареди ядрото на Linux директно от твърдия диск - виж следващия раздел.

Много дистрибуции създават дискета за зареждане на Linux по време на инсталирането на системата. При необходимост с тази дискета ще можете лесно да зареждате Linux. Така ще си спестите евентуални неприятности при зареждане на Linux от твърдия диск (понякога програмите за зареждане на OS/2 и Windows NT създават проблеми при конфигурирането им за зареждане на Linux - виж следващия раздел). След като ядрото се зареди, можете да използвате дискетното устройство за други цели.

Ако имате файл, в който е записано ядрото на Linux, можете да направите своя дискета за зареждане. В много Linux системи ядрото се намира във файла */boot/vmlimiz.* Все пак, това не е универсална конвенция - някои Linux системи записват ядрото във */vmlinuz* или */vmlinux,* а други използват файл като */Image.* (Ако имате няколко ядра с различни версии, можете да използвате LILO, за да изберете ядрото, което искате да се зареди - виж следващия раздел.) Някои току-що инсталирани Linux системи нямат копие от ядрото на твърдия диск, ако сте избрали само създаване на дискета за зареждане на Linux. При всички случаи, можете да създадете (тоест да компилирате) свое собствено ядро. Така или иначе това е полезно - можете да "конфигурирате" ядрото така, че да включва само тези драйвери, които са необходими за хардуера в компютъра ви (виж раздел "Създаване на ново ядро" в Глава 7).

Глава 7

Да приемем, че файлът */boot/vmlinuz* съдържа копие от ядрото. За да създадете дискета за зареждане на Linux, първо трябва да зададете кой дял на твърдия диск съдържа основната файлова система. (Ако сте компилирали ядрото, вече сте задали подходящата стойност, но тук можете да я проверите.) За целта се използва програмата *rdev.* Описахме как можете да създадете основната файлова система на Linux в разделите "Дискови устройства и техните дялове под Linux" и "Създаване на дялове за Linux" в Глава 3, *Инсталиране и начално конфигуриране.*

Глава 3

За да видите как се използва *rdev,* въведете *rdev -h* (преди това трябва да сте влезли като root). Ще откриете, че командата предоставя множество опции, с които можете да зададете къде се намира основната файлова система (което искаме да направим), мястото за swap, размера на вирту-

* Откъде идва това глупаво име? В много Unix системи ядрото се намира във файла */vmitiiix,* където ***vm*** идва от "virtual machine", т.е. виртуална машина. Разбира се, при Linux нещата трябва да са различни, затова името на ядрото става *vmlinux* и ядрото се записва в директорията */boot.* Името *vmlinuz* се използва, за да се различават компресираните от некомпресираните ядра. Всъщност, името и мястото на ядрото нямат никакво значение, ако зареждате от дискета или използвате програмата LILO, която знае точно къде се намира ядрото.

алните дискове (ramdisk) и т.н. Повечето от тези опции в момента не са ви необходими.

Ако изпълните командата *rdev ft>oot/vmlinuz,* ще видите името на устройството, което е кодирано в ядрото като място за основната файлова система:

```
courgette:/# rdev /boot/vmlinuz
Root device /dev/hda1
```

Ако вашата основна файлова система всъщност се намира в */devfhda3,* трябва да изпълните следната команда:

```
courgette: /# rdev /boot/vm3.inuz /dev/hda3
courgette:/#
```

rdev е -мълчалива програма и не отпечатва нищо, когато зададете устройството, в което се намира основната файлова система. Препоръчваме ви да изпълните още веднъж *rdev /boot/vmlinuz,* за да се убедите, че всичко е наред.

Вече сме готови за създаването на дискета за зареждане на Linux. За да нямате проблеми, използвайте нова, току-що форматирана дискета. Можете да форматирате дискетата под MS-DOS с командата FORMAT или под Linux с командата *fdformat.* Форматиращата програма ще запише информация за секторите и пътеките на дискетата, която се използва за автоматичното разпознаване на размера на дискетата (за повече информация виж раздел "Управление на файловите системи" в Глава 6).

За да създадете дискета за зареждане на Linux, използвайте командата *dd* по следния начин:

```
courgette:/# dd if=/boot/vmlinuz of=/dev/fd0 bs=8192
```

Ако се интересувате как работи командата *dd,* можете да прочетете справочната й страница, в която ще намерите множество примери. Накратко, горната команда копира файла с име */boot/vmlinuz* (задава се с опцията *if,* която идва от input file, тоест "входен файл") във файла с име */dev/fdO,* тоест първото дискетно устройство (задава се с опцията *of* която идва от output file, тоест "изходен файл"), като използва блокове с размер 8192 байта (опцията *bs* - идва от block size, тоест "размер на блока"). Разбира се, за тази цел можехме да използваме и плебейската команда *cp,* но ние, системните администратори на Unix, обичаме да използваме сложни команди, когато трябва да извършим сравнително прости неща. Точно това ни различава от обикновените смъртни.

*

В дистрибуцията на Debian няма команда *fdformat,* вместо нея можете да използвате командата с недвусмисленото име *superformat.*

Дискетата за зареждане на Linux би трябвало вече да е готова. Можете да спрете системата си (виж раздела "Спиране на системата" по-долу в тази глава), след което да заредите ядрото от новата си дискета, и ако всичко върви добре, Linux ще тръгне както обикновено. Би било добре за всеки случай да имате такава резервна дискета за зареждане на Linux. В раздела "Какво да правим в аварийна ситуация" в Глава 8 ще научите как можете да използвате тази дискета при възстановяване на системата след срив.

Глава 8

Как да използваме LILO

LILO е програма за управление на началното зареждане (boot manager). Когато използвате LILO, можете да определяте коя операционна система да се зарежда при стартирането на компютъра - това може да е Linux или произволна друга операционна система. Съществуват много начини за конфигуриране на LILO; тук ще опишем само двата най-разпространени начина - инсталиране на LILO в първия сектор на диска (така наречения MBR, или основен запис за начално зареждане) и инсталиране на LILO в дяла на Linux.

LILO е най-разпространената програма за зареждане на Linux от твърд диск (под "зареждане от твърд диск" разбираме това, че самото ядро е записано на твърдия диск и не се налага използване на дискета за зареждане на Linux; трябва да отбележим, че дори при зареждане от дискета, управлението се прехвърля на твърдия диск, след като ядрото се зареди в паметта). Ако инсталирате LILO в първия сектор на диска си (тоест MBR), това ще бъде първата програма, която ще се изпълнява при всяко зареждане от твърд диск. След като тръгне LILO, можете да изберете коя операционна система искате да се зареди - например Linux или MS-DOS - и LILO ще я зареди.

OS/2 и Windows NT използват свои собствени програми за управление на началното зареждане, които заемат MBR. Ако използвате една от тези системи и искате да зареждате Linux от твърдия диск, можете да инсталирате LILO в първия сектор на дяла, който съдържа основната файлова система на Linux. Програмата за управление на началното зареждане (от OS/2 или Windows NT) се грижи за стартирането на LILO, когато искате да заредите Linux.

За съжаление, въпросният софтуер от OS/2 и Windows NT понякога създава проблеми при стартиране на LILO. Причината за това е лошия му дизайн и ако наистина се налага да го използвате, може би ще бъде полезно да зареждате Linux от дискета.

От друга страна, LILO работи безпроблемно с Windows 95/98. Конфигурирането на LILO за зареждане на Windows 95/98 не се различава от ко-

нфигурирането му за зареждане на MS-DOS (виж следващия раздел). Все пак, ако инсталирате Windows 95/98 след като вече сте инсталирали LILO, ще ви се наложи да преинсталирате LILO (защото инсталационната процедура на Windows 95/98 променя MBR на твърдия диск). Достатъчно е да имате подготвена дискета за зареждане на Linux, от която да заредите операционната система, след което ще можете да преинсталирате LILO.

Преди да продължим, трябва да отбележим, че много от дистрибуциите на Linux по време на инсталирането си могат автоматично да конфигурират и инсталират LILO. Все пак ви препоръчваме лично да конфигурирате LILO, просто за да сте сигурни, че всичко е наред.

Файлът /etc/lilo.conf

Първата стъпка в конфигурирането на LILO е създаването и редактирането на конфигурационния файл на LILO. Обикновено този файл е */etc/lilo.conf* (в някои системи файлът е *fooot/lilo.conf* или */etc/lilo/config).*

Ще опишем един примерен конфигурационен файл. Можете да го използвате като основа и да го промените по подходящ начин за вашата система.

Първата част от файла задава някои основни параметри:

```
boot    = /dev/hda
compact
install = /boot/boot.b
map     = /boot/map
```

Първият ред (boot) задава името на устройството, в което LILO ще се инсталира като запис за начално зареждане. В този случай искаме да инсталираме LILO в първия сектор (тоест MBR) на */dev/hda* - първият твърд диск с IDE интерфейс. Ако зареждате от SCSI диск, използвайте съответното име, например */dev/sda.* Ако искате LILO да зарежда само Linux, трябва да зададете името на дяла с основната файлова система на Linux (например */dev/hda2).* По-късно ще обясним този вариант по-подробно.

Вторият ред (compact) указва на LILO да извърши някои оптимизации; можете да използвате тази опция винаги, освен ако не се занимавате сериозно с конфигурацията на LILO. Препоръчваме ви да използвате следващите два реда (install и map) по показания тук начин, install

В някои случаи се налага да използвате опцията **linear,** която не трябва да се използва едновременно с **compact.** За повече информация виж документацията на LILO.

задава името на файла, съдържащ програмния код, който се записва в MBR; map задава името на файла, в който са описани секторите на диска, които съдържат ядрото на Linux. Обикновено тези файлове се намират в директорията */boot,* но при някои системи те могат да бъдат в */etc/lilo.* Файлът */boot/map* се създава, когато LILO се инсталира за първи път.

За всяка операционна система, която искате да се стартира от LILO, трябва да добавите по една секция към */etc/lilo.conf.* Например за Linux можете да добавите следното:

```
# Конфигурация за Linux с основна файлова система в /dev/hda2.
    image = /boot/vmlinuz   # файл, който съдържа ядрото
    label = linux           # името на ОС в менюто на LILO
    root  = /dev/hda2       # място на основната файлова система
    vga   = ask             # текстов режим за системната конзола
```

С параметъра image задавате името на файла, който съдържа ядрото на Linux. С label задавате името на конфигурацията в менюто на LILO при начално зареждане. Следват параметрите root - задавате кой дял съдържа основната файлова система на Linux, и vga - задавате кой текстов режим на видеоконтролера VGA ще се използва за системната конзола.

Валидните стойности за параметъра vga са normal (стандартен дисплей 80x25 символа); extended (за текстов режим с повече символи, обикновено 132x44 или 132x60); ask (за въвеждане на стойност при зареждането на системата) или цяло число (1, 2 и т.н.). Цялото число е номера на валиден текстов режим. Какви текстови режими можете да използвате зависи от видео платката в компютъра ви; можете да използвате vga = ask, за да получите списъка им.

Ако използвате няколко версии на ядрото на Linux - например когато опитвате нова версия на ядрото - можете да добавите по една секция за всяка версия. В този случай единственият задължителен параметър освен image е label. Ако не зададете параметрите root или vga, ще се използват стойностите, кодирани в ядрото с програмата *rdev.* Ако зададете параметри root или vga, техните стойности ще заменят подразбиращите се стойности, които сте задали с *rdev.* Затова, ако зареждате Linux с LILO, няма необходимост да използвате *rdev;* можете да зададете желаните параметри в конфигурационния файл на LILO.

Секцията, с която задавате дефиниция за зареждане на MS-DOS или Windows 95/98 би могла да изглежда по следния начин:

```
# Конфигурация за зареждане на операционна система
# от дял на MS-DOS/Win 95/Win 98
    other = /dev/hda1       # Място на дяла
    table = /dev/hda        # Място на таблицата с дяловете
    label = msdos           # Име на ОС  (за менюто на LILO)
```

Бихте могли да използвате аналогична секция за зареждане на OS/2 (разбира се, след като смените името на операционната система).

Ако искате да заредите от MS-DOS или Windows 95/98 дял, който се намира във втория твърд диск, трябва да добавите следния ред:

```
loader - /boot/any_d.b
```

в секцията за MS-DOS. Ако искате да заредите OS/2 от втория твърд диск, трябва да добавите реда:

```
loader = /boot/os2_d.b
```

Съществуват още много опции за конфигуриране на LILO. Самата дистрибуция на LILO (може да се намери в повечето дистрибуции и FTP сайтове, свързани с Linux) включва подробно ръководство, в което можете да намерите описание на всички възможни опции. Все пак, за повечето системи е достатъчно да използвате опциите в горните примери.

След като подготвите конфигурационния файл */etc/lilo.conf,* можете да инсталирате LILO с командата:

```
/sbin/lilo
```

Горната команда трябва да се изпълни като root. Би трябвало инсталационната команда да изведе следната информация:

```
courgette:/# /sbin/lilo
Added linux
Added msdos
courgette:/#
```

Ако зададете опцията -v, */sbin/lilo* ще изведе по-подробна информация, която може да се използва, ако възникнат проблеми; опцията -C ви позволява да използвате друг конфигурационен файл вместо стандартния */etc/lilo.conf*.

След като инсталирате LILO, можете да спрете системата си (още веднъж ви напомняме да прочетете раздела "Спиране на системата" по-долу в тази глава), да рестартирате компютъра и да изпробвате как работи LILO. По подразбиране се зарежда операционната система, описана в първата секция на */etc/lilo.conf*. За да изберете друго ядро или друга операционна система, задръжте натиснат клавиша Shift или клавиша Control (или просто включете Scroll Lock) при зареждането на Linux. Би трябвало да получите поканата на LILO при зареждане на системата:

```
boot:
```

Можете да натиснете клавиша Tab, за да получите списък с дефинираните операционни системи:

```
boot: натиснете-клавиша-Tab
linux msdos
```

Това са имената на операционните системи, които сте задали чрез параметрите **label** в */etc/lilo.conf*. Въведете името на избраната от вас операционна система и LILO ще я зареди. В този случай, ако въведете **msdos,** LILO ще зареди MS-DOS от */dev/hdal,* както е зададено във файла ***/etcflilo.conf.***

Инсталиране на LILO в дяла на Linux

Ако използвате програма за управление на началното зареждане (boot manager) от OS/2 или Windows NT, инсталирате Debian или просто не искате LILO да се намира в първия сектор (MBR) на диска ви, можете да инсталирате LILO в дяла с основната файлова система на Linux.

Можете да направите това, като промените параметъра **boot** = ... в */etc/lilo.conf,* така че да съдържа името на дяла на Linux. Например редът:

```
boot = /dev/hda2
```

означава, че LILO трябва да се инсталира в първия сектор на */dev/hda2.* Забележка: това е валидно само за основни (primary) дялове на твърдия диск (но не и за "разширен" (extended) - или логически дял). Ограничението се отнася преди всичко за дистрибуцията на Debian; в тази дистрибуция се разчита на програмния код в MBR, а той не може да зареди операционна система от логически дял на диска. Ако използвате дистрибуцията на Debian и искате да заредите от твърд диск, трябва да маркирате дяла на Linux като "активен" в таблицата с дяловете на диска. Това може да се направи с *fdisk* под Linux или MS-DOS. Когато стартирате компютъра, ще се зареди операционната система в "активния" дял на диска, тоест Linux.

Ако използвате boot manager от OS/2 или Windows NT, трябва да инсталирате LILO в дяла на Linux, след което да конфигурирате съответния boot manager да зарежда операционната система от този дял на диска. Как точно се прави това, можете да разберете от документацията на вашия boot manager.

J>s

Ш

LILO може да работи с boot manager от OS/2, но това не винаги се постига лесно. Проблемът е в това, че OS/2 boot manager дори не разпознава дяла, създаден ***c/disk*** под Linux. Понякога можете да заобиколите проблема като използвате *fdisk* под OS/2, за да дадете име на дяловете на Linux (създадени с *fdisk* под Linux). Друго решение на проблема е да създадете дяловете за Linux с *fdisk* под OS/2 (например като дялове за MS-DOS). В такъв случай OS/2 ще разпознае дяловете, след което с *fdisk* под Linux можете да промените типа им на **Linux native** и **Linux swap** по начина, описан в раздела "Създаване на дялове за Linux" в Глава 3. След това в тези дялове можете да инсталирате Linux, а LILO трябва да

Глава 3

се инсталира в дяла с основната файлова система. Ако имате късмет, това е всичко.

Защо ви казваме всичко това? Защото OS/2 Boot Manager не може да зарежда операционни системи, които не разпознава. Препоръчваме ви да инсталирате LILO в MBR на диска си и да го конфигурирате да зарежда OS/2. Това става като добавите секция за OS/2 в */etc/lilo.conf* - аналогично на секцията за MS-DOS. Друга възможност е да зареждате Linux от дискета, или просто да не използвате OS/2. Но нека не се отклоняваме.

Задаване на параметри при зареждане на Linux

Когато за пръв път инсталирате Linux, вероятно ще заредите от дискета или компакт-диск и ще получите вече познатия ви интерфейс на LILO при начално зареждане. Можете да въведете различни опции като:

```
Ъй=цилиндри,глави,сектори
```

за да зададете геометрията на диска си. Понякога се налага да задавате тези параметри всеки път, когато зареждате Linux, за да може ядрото да разпознае коректно хардуера в компютъра ви (виж раздела "Зареждане на Linux" в Глава 3). Ако използвате LILO за зареждане на Linux от твърдия диск, можете да зададете тези параметри в */etc/lilo.conf,* вместо да ги въвеждате ръчно при всяко зареждане на Linux. В секцията за Linux в */etc/lilo.conf* трябва да добавите нов ред, например:

```
append = "hd=683,16,38"
```

Така системата ще работи по същия начин, както би работила, ако въвеждате **hd=683, 16,38** при всяко зареждане на Linux. Ако искате да зададете няколко опции, можете да използвате един ред, например:

```
append - "hd=683,16,38 hd=64,32,202"
```

В този случай задавате геометрията съответно на първия и на втория твърд диск.

Трябва да използвате тези опции, само ако при зареждане на Linux ядрото не разпознава хардуера в компютъра ви. Ако имате опит с инсталирането на Linux, вече знаете дали е необходимо да въвеждате такива параметри; в общия случай трябва да използвате реда append в *lilo.conf* само ако се е наложило да въведете тези опции при първото зареждане на Linux от инсталационния носител.

Съществуват още няколко параметъра, които можете да използвате при зареждане на Linux. Повечето от тях се използват за разпознаването на хардуер, което обсъдихме в Глава 3. Все пак, има още няколко опции, които могат да ви бъдат полезни:

single

Зарежда системата в еднопотребителски режим без да се изпълняват конфигурационните файлове. Използва се от системния администратор (виж раздела "Какво да правим в аварийна ситуация" в Глава 8).

ffl

Глава 8

root=дял

Монтира зададения дял като основна файлова система. Тази опция има приоритет пред стойността, зададена във файла */etc/lilo.conf.*

ro Монтира основната файлова система в режим, в който е разрешено само четенето от нея. Обикновено това се прави, за да може по-късно да се стартира програмата^сА: (виж раздел "Проверка и поправка на файловите системи" в Глава 6).

Глава 6

ramdisk=pa^j*ep

Задава размера (в байтове) на виртуалния диск. Тази опция има приоритет пред стойността, зададена във файла */etc/lilo.conf.* Обикновено виртуалният диск се използва при инсталирането на Linux.

vga=ред:ММ

Задава текстовия режим на видео контролера VGA. Тази опция има приоритет пред стойността, зададена във файла */etc/lilo.conf.* Валидните стойности са **normal, extended, ask** или цяло число (виж раздела "Файлът /etc/lilo.conf по-горе в тази глава).

тет=размер

Задава количеството физическа памет в компютъра. Ако разполагате с по-малко от 64 MB RAM, ядрото може да получи тази информация от системния BIOS. Но ако имате повече памет и използвате стара версия на ядрото, трябва да зададете на ядрото точното количество памет в компютъра си, защото в противен случай ядрото ще използва само 64 MB. Например, ако имате 128 MB, трябва да зададете **mem=12 8m.** Тази опция не е необходима, ако използвате нова версия на ядрото.

Всяка от тези опции може да бъде въведена ръчно в интерфейса на LILO при начално зареждане или да бъде зададена с **append** в *etc/lilo.conf.*

Пакетът LILO включва пълна документация, в която са описани всички конфигурационни опции. В много Linux системи тази документация се намира в */usr/src/lilo;* в Debian можете да я намерите в */usr/doc/lilol Manucil.txt.gz.* Ако не можете да намерите документацията в системата си, можете да я изтеглите от някой Интернет-сайт с информация за Linux. Документацията включва ръководство, в което детайлно са описани принципите за зареждане и използване на LILO, както и текстов файл *{README),* който съдържа по-важните части от това ръководство.

Премахване на LILO

Ако сте инсталирали LILO в първия сектор на диска си (MBR), най-лесният начин да го премахнете е да използвате MS-DOS командата FDISK. Командата:

```
FDISK /MBR
```

стартира FDISK и записва на мястото на LILO стандартния запис за начално зареждане на MS-DOS.

LILO записва резервно копие на оригиналния запис за начално зареждане във файловете *foootfi>oot.0300* (за IDE дискове) и */boot^boot.0800* (за SCSI дискове). Тези файлове съдържат програмния код в MBR на диска преди инсталирането на LILO. Можете да използвате командата *dd,* за да замените LILO с оригиналния запис за начално зареждане. Например командата:

```
dd if=/boot/boot.0300 of=/dev/hda bs=446 count=1
```

копира първите 446 байта от файла *ft>oot/boot.0300* в устройството */dev/hda.* Дори ако файловете са с размер 512 байта, трябва да копирате само първите 446 от тях в MBR.

Внимавайте когато изпълнявате тази команда! Това е един от случаите, в които ако изпълните без много-много да мислите команда, намерена в иначе добра книга, можете да си причините истински проблеми (особено, ако не знаете какво правите). Използвайте този метод като последно средство и то само ако сте напълно сигурни, че файловете */bootft>oot.0300* или *fooot/boot.0800* съдържат записа за начално зареждане, който искате да използвате. Много дистрибуции на Linux се разпространяват с фиктивни данни в тези два файла; най-добре е да ги изтриете, преди да инсталирате LILO.

Документацията на LILO съдържа още няколко начина за премахване на LILO и друга полезна информация.

Инициализация на системата

В този раздел ще опишем детайлно действията, които се извършват при зареждането на системата. За всеки системен администратор е важно да разбира този процес и да знае кои файлове се използват при зареждане на системата.

Съобщения на ядрото при зареждане на системата

Първата стъпка е зареждането на ядрото. Както описахме в предишния раздел, ядрото може да се зареди от дискета или от твърд диск. След като ядрото се зареди в паметта, то отпечатва съобщения на системната конзола, които обикновено се записват и в log-файловете. Влезте като root и разгледайте файла */var/log/messages* (който съдържа всички съобщения, отпечатани по време на работата на ядрото). Командата *dmesg* отпечатва последните редове от кръговия буфер със съобщения на ядрото. Разбира се, ако изпълните *dmesg* веднага след зареждане на системата, ще получите съобщенията на ядрото при зареждане на системата (възможно е съобщенията на вашето ядро да не са точно същите, нито в същия ред):

```
Console: 16 point font, 480 scans
Console: colour VGA+ 80x30, 1 virtual console (max 63)
pcibios_init : BIOS32 Service Directory structure at 0x000fb1d0
pcibios_init : BIOS32 Service Directory entry at 0xfb5a0
pcibios_init : PCI BIOS revision 2.00 entry at 0xfb5d0
Probing PCI hardware.
Calibrating delay loop.. ok - 36.04 BogoMIPS
Memory: 14984k/16384k available (552k kernel code, 384k reserved,\
464k data)
Swansea University Computer Society NET3.035 for Linux 2.0
NET3: Unix domain sockets 0.13 for Linux NET3.035.
Swansea University Computer Society TCP/IP for NET3.034
IP Protocols: ICMP, UDP, TCP
VFS: Diskquotas version dquot_5.6.0 initialized
Checking 367/387 coupling... Ok, fpu using exception 16 error\
reporting.
Checking 'hlt' instruction... Ok.
Intel Pentium with F0 0F bug - workaround enabled.
alias mapping IDT readonly. . . . . done
Linux version 2.0.35 (root@rabbit) (gcc version egcs-2.90.29\
980515 (egcs-1.0.3 release)) #3 Fri Nov 13 15:07:45 CET 1998
Starting kswapd v 1.4.2.2
Serial driver version 4.13 with no serial options enabled
tty00 at 0x3f8 (irq = 4) is a 16550A
tty01 at 0x2f8 (irq = 3) is a 16550A
APM BIOS not found.
Real Time Clock Driver v1.09
Configuring Adaptec (SCSI-ID 7) at IO:330, IRQ 11, DMA priority 5
scsi0 : Ataptec 1542
scsi : 1 host.
  Vendor: IBM      Model: DORS-32160       Rev: WA0A
  Type:   Direct-Access                    ANSI SCSI revision: 02
Detected scsi disk sda at scsi0, channel 0, id 0, lun 0
  Vendor: SANYO    Model: CRD-254S         Rev: 1.05
```

```
  Type:   CD-ROM                              ANSI SCSI revision: 02
Detected scsi CD-ROM sr0 at scsi0, channel 0, id 4, lun 0
scsi : detected 1 SCSI cdrom 1 SCSI disk total.
SCSI device sda: hdwr sector= 512 bytes. Sectors= 4226725 [2063 MB]\
[2.1 GB]
Partition check:
 sda: sda1 sda2 sda3
VFS: Mounted root (ext2 filesystem) readonly.
Adding Swap: 130748k swap-space (priority -1)
lp1 at 0x0378, (polling)
3c59x.c:v0.99E 5/12/98 Donald Becker http://cesdis.gsfc.nasa.gov/\
linux/drivers/vortex.html
eth0: 3Com 3c905B Cyclone 100baseTx at 0x6000, 00:10:4b:45:1d:53,\
IRQ 12
 8k byte-wide RAM 5:3 Rx:Tx split, 10baseT interface.
 Enabling bus-master transmits and whole-frame receives.
ISDN subsystem Rev: 1.44/1.41/1.47/1.28/none loaded
HiSax: Driver for Siemens chip set ISDN cards
HiSax: Version 2.1
HiSax: Revisions 1.15/1.10/1.10/1.30/1.8
HiSax: Total 1 card defined
HiSax: Card 1 Protocol EDSSI Id=teles (0)
HiSax: Teles IO driver Rev. 1.11
HiSax: Teles 16.3 config irq:5 isac:980  cfg:d80
HiSax: hscx A:180  hscx B:580
Teles3: HSCX version A: V2.1  B: V2.1
Teles3: ISAC 2086/2186 V1.1
HiSax: DSS1 Rev. 1.16
HiSax: 2 channels added
HiSax: module installed
inserting floppy driver for 2.0.35
Floppy drive(s): fd0 is 1.44M
FDC0 is an 8272A
```

Тези съобщения се отпечатват от самото ядро по време на инициализирането на драйверите на хардуерните устройства. Точно какви съобщения се отпечатват, зависи от това, какви драйвери са включени в ядрото и какъв хардуер имате в компютъра си. Следва кратко описание на по-важните съобщения и значението им.

Първо, ядрото съобщава кой шрифт използва за системната конзола и какъв е типа на конзолата. Забележка: това се отнася само за текстовия режим, който се използва от ядрото, а не за възможностите на видео картата (всяка SVGA видео карта се смята за **VGA+,** що се отнася до текстовия режим, който се използва за конзолата).

След това ядрото събира информация за PCI шината и PCI устройствата в компютъра.

Следващото съобщение е за изчисляването на "BogoMIPS" за вашия процесор. Това е абсолютно фалшива (Bogo идва от "bogus", тоест фал-

шив) мярка за скоростта на процесора ви и се използва само за получаване на оптимална производителност при някои драйвери. Освен това, ядрото отпечатва информация за количеството на паметта в компютъра ви:

```
Memory: 14984k/16384k available (552k kernel code, \
384k reserved, 464k data)
```

Виждаме, че за приложенията остават 14984 KB памет. Това означава, че самото ядро използва 1500 KB.

След това се инициализира кодът за поддръжка на мрежа и се проверява типът на централния процесор. От реда

```
Intel Pentium with F0 0F bug - workaround enabled.
```

можете да видите, че ядрото на Linux е достатъчно умело, за да се справи с позорната грешка при изчисления с плаваща запетая в ранните Pentium-и. Редът:

```
Linux version 2.0.35 (root0rabbit) (gcc version \
egcs-2.90.29 (980515 (egcs-1.0.3 release)) \
#3 Fri Nov 13 15:07:45 CET 1998
```

показва версията на ядрото; потребителят, който го е компилирал и машината, на която е извършена компилацията (в този случай root на машината rabbit) и накрая коя версия на компилатора е използвана. Следва информация за всеки открит сериен порт от съответния драйвер. Редът

```
tty00 at 0x3f8 (irq = 4) is a 16550A
```

означава, че първият сериен порт *(/dev/tty00* или COM1 в MS-DOS) използва порт Ox03f8, LRQ 4 и е съвместим със стандартния UART 16550A. Следва проверката и конфигурирането на SCSI контролерите в машината. Ядрото отпечатва информация за всяко открито SCSI устройство. Редът

```
Adding Swap: 130748k swap-space (priority -1)
```

показва какво количество swap ще се използва за виртуална памет. След това ядрото открива и конфигурира паралелния порт (lp1), мрежовата карта и накрая настройва подсистемата си за работа с ISDN. Последното съобщение на ядрото е за откриването на дискетното устройство. В зависимост от хардуера в компютъра ви, ядрото може да изведе още съобщения за открити и конфигурирани устройства, например за допълнителни SCSI контролери.

Програмата init, файлът inittab и гс-скриптовете

След като инициализира драйверите на устройствата в компютъра ви, ядрото изпълнява програмата *init,* която се намира в */etc, fl>in* или */sbin* (в повечето системи файлът е */sbin/inii). init* е специална програма, която пуска нови процеси или в някои ситуации рестартира определени програми. Например, за всяка виртуална конзола работи един процес *getty,* който е пуснат от *init.* Когато завършите сесията си на дадена виртуална конзола (тоест излезете от системата), процесът *getty* прекратява изпълнението си, след което *init* пуска нов процес *getty,* за да можете отново да влезете в системата.

При зареждането на системата *init* стартира няколко програми и скриптове на командния интерпретатор. Всяка дейност на *init* се управлява от файла */etc/inittab.* Всеки ред в този файл има следния формат:

```
code:runlevels:action:command
```

code (код) е уникална последователност от един или два символа и се използва за идентифициране на конкретния ред от файла. Някои редове трябва да имат определен код, за да работят правилно (виж по-долу).

runlevels е списък от тези нива на работата на системата, при които трябва да се изпълни командата в реда. Например, когато нивото на работа стане 3, ще бъдат изпълнени командите във всички редове на файла */etc/inittab,* които съдържат цифрата 3 в полето *runlevels.* От гледна точка на *init,* нивото на работа е число или буква, отговарящо на определено състояние на системата. Използването на нива на работа е лесен начин за групиране на редовете в */etc/inittab.* Например, бихте могли да зададете следните действия: при ниво 1 да се изпълнят само най-необходимите конфигурационни скриптове; при ниво 2 да се изпълнят действията при ниво 1 и конфигурацията на мрежовата подсистема на Linux; при ниво 3 да се изпълнят действията при нива 1 и 2 плюс стартиране на web-сървър и т.н. Съвременните дистрибуции на Red Hat са настроени така, че при ниво на работа 5 автоматично се стартира графичният интерфейс на Linux (тоест системата X Window). Дистрибуцията на SuSE извършва същото при ниво 3, а тази на Debian - при ниво 2 (разбира се, преди това трябва да сте инсталирали X).

За повечето потребители не са важни конкретните детайли за употребата на нивата на работа. Когато системата се зареди, тя влиза в подразбиращо се ниво на работа (задава се в */etc/inittab,* както ще видим след малко), като в повечето системи това е 2-ро или 3-то ниво. След като разгледаме нормалната процедура за зареждане на Linux, ще ви покажем как се влиза в едно друго ниво на работа, което понякога ще използвате - runlevel 1, или еднопотребителски режим на работа.

Нека разгледаме един примерен файл */etc/inittab:*

```
# Подразбиращото се ниво на работа ще бъде 3
id:3:initdefault:

# Изпълнява /etc/rc.d/rc.sysinit при зареждането на системата
si:S:sysinit:/etc/rc.d/rc.sysinit

# Стартира /etc/rc.d/rc с аргумент нивото на работа
10:0:wait:/etc/rc.d/rc 0
11:1:wait:/etc/rc.d/rc 1
12:2:wait:/etc/rc.d/rc 2
13:3:wait:/etc/rc.d/rc 3
14:4:wait:/etc/rc.d/rc 4
15:5:wait:/etc/rc.d/rc 5
16:6:wait:/etc/rc.d/rc 6

# Изпълнява се, когато потребителят натисне Ctrl-Alt-Delete
ca::ctrlaltdel:/sbin/shutdown -t3 -rf now

# Стартира програмата agetty за виртуални конзоли с номера от 1 до 6
c1:12345:respawn:/sbin/agetty 38400 tty1
c2:12345:respawn:/sbin/agetty 38400 tty2
c3:45:respawn:/sbin/agetty 38400 tty3
c4:45:respawn:/sbin/agetty 38400 tty4
c5:45:respawn:/sbin/agetty 38400 tty5
c6:45:respawn:/sbin/agetty 38400 tty6
```

Полетата се отделят едно от друго с двоеточие. Най-ясен е смисълът на последното поле - това е програмата, която *init* ще стартира, ако са изпълнени съответните условия. Първото поле представлява произволен идентификатор (името му няма значение, стига да е уникално във файла). Второто поле е нивото на работа (runlevel), при което се изпълняват отбелязаните в реда действия. Третото поле задава начина, по който *init* трябва да обработи реда - например, дали да изпълни зададената команда или да я стартира отново (respawn), когато тя завърши работата си.

Точното съдържание на файла */etc/inittab* зависи от компютъра ви и дистрибуцията на Linux, която сте инсталирали.

В първия ред на нашия примерен файл виждаме, че подразбиращото се ниво на работа се установява на 3. Полето **`action`** за този ред съдържа `initdefault`, което означава, че подразбиращото се ниво на работа ще бъде стойността, зададена в полето *ninlevel* на този ред. Това е нивото, което обикновено се използва при нормална работа на системата. По-късно можете да смените това ниво с произволно избрано от вас ниво, като стартирате ръчно *init* и зададете желаната стойност като аргумент (можете да използвате тази възможност, когато коригирате грешките в конфигурацията си). В следващия пример командата *init* ще прекрати работата на всички работещи процеси и ще превключи на ниво на работа 5 (първо предупредете всички работещи потребители да напуснат системата!):

```
tigger# init 5
```

Забележка: LILO може да зареди Linux в еднопотребителски режим (обикновено това е ниво на работа 1) - виж раздела "Задаване на параметри при зареждане на Linux" по-горе в тази глава.

Следващият ред в примерния файл *inittab* ще изпълни скрипта */etc/rc..d/rc.sysinit* при зареждането на системата. (Полето *action* съдържа sysinit, което означава, че този ред трябва да се изпълни, когато *init* се стартира за пръв път - веднага след като ядрото завърши зареждането си). Този файл е просто скрипт на командния интерпретатор, който съдържа команди, с които се извършва базовата системна инициализация - например, активира се виртуалната памет, проверяват се и се монтират файловите системи, системният часовник се синхронизира с часовника в CMOS паметта на компютъра. Можете да хвърлите един поглед върху този файл в системата си; ще разгледаме подробно командите в него в Глава 6 (виж разделите "Управление на файловите системи" и "Управление на виртуалната памет"). При някои дистрибуции файлът има друго име, например при SuSE това е */sbiii/init.d/boot.*

Глава 6

По-долу в нашия пример виждаме, че системата изпълнява скрипта */etc/rc.d/rc,* когато влезе в ниво на работа от 0 до 6, като задава нивото на работа като аргумент на скрипта. *rc* е скрипт с общо предназначение и се използва за стартиране на други скриптове, в зависимост от съответното ниво на работа. Полето *action* съдържа wait, което означава, че *init* трябва да изпълни зададената команда и да изчака тя да завърши, преди да започне друго действие.

В Linux командите, които трябва да се изпълнят при зареждане на системата, се съхраняват във файлове, съдържащи в името си *rc.* Тези команди извършват всички необходими действия, за да получите една напълно функционираща система, например стартират демоните, които споменахме в Глава 4, *Основни команди и концепции в Unix.* Благодарение на тези команди след стартирането на Linux работят системи, които реализират електронна поща, web-сървър или други услуги, които сте инсталирали на компютъра си. rc-файловете се стартират от файла */etc/inittab* и представляват стандартни скриптове на командния интерпретатор; можете да разгледате командите в тях с произволен текстов редактор.

Глава 4

В този раздел ще опишем структурата на rc-файловете, така че да разберете къде се стартират различните процеси и да можете ръчно да пускате или спирате сървъри в редките случаи, когато системата не прави това, което искате. Ще използваме като модел дистрибуцията на Red Hat, но след като получите обща представа за структурата на rc-файловете, ще можете да откриете необходимите ви файлове в произволна дистрибуция на Linux.

В Red Hat rc-скриптът от най-горно ниво е */etc/rc.d/rc*. В някои дистрибуции се използва друг път (например */etc/init.d/rc* в Debian), но съдържанието на файла е почти същото. В предишния раздел можете да видите как */etc/inittab* използва този скрипт в разнообразни ситуации, като задава числата от 0 до 6 като аргументи. Тези числа съответстват на нивата на работа (runlevels), като за всяко число скриптът от най-горно ниво изпълнява различни групи скриптове от по-ниско ниво. Следователно, следващата ни стъпка е да открием скриптовете, които съответстват на различните нива на работа.

В Red Hat скриптовете за всяко ниво на работа (runlevel) се съхраняват в директорията */etc/rc.d/rcN.d,* където N е номера на нивото на работа. Например, за ниво на работа 3 се използват скриптовете в директорията */etc/rc.d/rc3.d*. Още веднъж ще споменем, че в други дистрибуции тази конвенция е леко изменена. Например, в Debian директорията за всяко ниво за работа е */etc/rcN.d*.

Хвърлете един поглед на някоя от тези директории; ще видите известно количество файлове с имена от вида ***snnxxxx*** или ***Knnxxxx,*** където nn е число от 00 до 99, а ***xxxx*** е името на услуга, която се предлага от системата, */etc/rc.d/rc* изпълнява първо скриптовете с имена ***Knnxxxx,*** за да спре някои работещи услуги (буквата K идва от "kill", тоест убивам), след което изпълнява скриптовете с имена ***Snnxxxx,*** за да пусне нови услуги (буквата S идва от "start", тоест стартирам).

Числата *nn* се използват, за да се зададе последователността, в която се изпълняват скриптовете - първо се изпълняват скриптовете, които имат по-малък номер. Името ***xxxx*** помага на системния администратор да идентифицира услугата, която се активира от съответния скрипт. Може би конвенцията за имената на rc-файловете изглежда странно, но тя улеснява добавянето и премахването на скриптове, както и автоматичното им изпълняване в подходящото време от */etc/rc.d/rc*. Съществуват програми с графичен интерфейс, с които можете да настройвате *rc*-скриптове в системата си - например *ksysv* за KDE (виж раздел "Работната среда KDE" в Глава 11, *Настройка на графичната среда).*

Глава 11

Два примера: скриптът, който инициализира мрежовата подсистема на Linux, може да се нарича *S10network,* а скриптът, който прекратява работата на log-демона, записващ системните съобщения, може да се нарича *K70syslog*. Ако тези файлове се поставят в подходящите директории */etc/rc.d/rcN.d,* скриптът */etc/rc.d/rc* ще ги изпълни в съответния ред по време на стартиране или спиране на системата. Ако подразбиращото се ниво на работа за вашата система е 3, погледнете съдържанието на директорията */etc/rc.d/rc3.d,* за да видите кои скриптове се изпълняват при нормално зареждане на системата ви.

Понеже едни и същи услуги се пускат или спират при различни нива на работа, дистрибуцията на Red Hat използва символни връзки, вместо да записва един и същи скрипт в няколко директории. Затова всеки rc-файл е символна връзка към скрипт в централна директория, която съдържа скриптовете за пускане и спиране на всички услуги в системата. В Red Hat тази директория е */etc/rc.d/init.d,* а в SuSE и Debian тя е */etc/init.d.* В Debian тази директория съдържа скрипт (нарича се *skeleton),* който можете да промените така, че да пуска и спира ваши собствени демони.

Полезно е да знаете къде се намират скриптовете за пускане и спиране на услугите в системата, например ако не искате да рестартирате компютъра или да променяте нивото на работа, когато се налага да спрете или пуснете конкретна услуга. Намерете скрипта с подходящо име в директорията *init.d,* след което го пуснете с параметър `start` или `stop.` Например в SuSE, ако искате да пуснете web-сървъра Apache в ниво на работа, при което той обикновено не работи, трябва да въведете следното:

```
tigger# /etc/init.d/apache start
```

Друг важен конфигурационен скрипт е */etc/rc.d/rc.local,* който се изпълнява след като се стартират другите инициализационни скриптове в системата. (Как се прави това? Обикновено във всяка директория */etc/rc.d/rcN.d* се създава символна връзка към *rc.local* с име *S99local.* 99 е най-голямото число, което може да се съдържа в името на rc-скрипт, и затова *S99local* се стартира последен. *Voila!).* В *rc.local* можете да добавите специфични системни команди или скриптове, за които не сте сигурни кога трябва да се изпълнят. Debian няма ^квивалент на скрипта *rc.local,* но нищо не ви пречи при необходимост.да го създадете и стартирате от скрипта *rc.*

Следващият ред в *inittab* има етикет са и се изпълнява, когато потребителят натисне клавишната комбинация Ctrl-Alt-Del в Някоя текстова конзола. Тази комбинация генерира прекъсване и обикновено се използва за рестартиране на системата. Ядрото на Linux "прехваща" това прекъсване и го изпраща на *init,* след което *init* изпълнява командата на реда, в който полето `action` съдържа `ctrlaltdel.` В нашия примерен файл командата е */sbin/shiitdown -t3 -rf now,* която ще рестартира системата по "правилния" начин (виж раздел "Спиране на системата" по-долу в тази глава).

И накрая, във файла *inittab* са включени няколко реда, които съдържат команди за изпълнение на */sbin/agetty* на първите шест виртуални конзоли, *agetty* е един от вариантите на командата *getty* под Linux. Програмите от този тип се използват за влизане в системата от терминал; без тях терминалът ще бъде практически мъртъв и няма да реагира на командите на потребителя. Различните варианти на *getty* инициализират терми-

нално устройство (например виртуална конзола, серийна линия и т.н.), задават разнообразни параметри за драйвера на терминала, след което изпълняват програмата */bin/login,* за да може потребителят да влезе в системата. Следователно, за да може потребителят да влезе в системата чрез дадена виртуална конзола, на нея трябва да работи *getty* или *agetty.* Повечето Linux системи използват варианта *agetty,* който има мНого близък синтаксис до този на *getty* (вижте справочната страница на *getty* или *agetty* на системата ви).

agetty използва два аргумента: скорост на връзката (baud rate) и име на устройство. В Linux устройствата, които отговарят на виртуалните конзоли, са */dev/ttyl, /dev/tty2* и т.н. *agetty* автоматично добавя */dev* пред аргумента "име на устройство". Скоростта на връзка за виртуалните конзоли обикновено е 38400.

Забележете, че полето *action* за всеки ред с *agetty* съдържа respawn.
Това означава, че ***init*** трябва да рестартира командата на този ред, след като процесът *agetty* умре, което става всеки път, когато потребителят излезе от системата.

Вече трябва да знаете как работи ***init.*** И все пак файловете и командите в */etc/rc.d,* които извършват по-голямата част от същинската работа, остават загадка. Не бихме могли да изследваме задълбочено тези файлове, преди да ви дадем повече информация за администрирането на системата, например за управлението на файловите системи. Ще разгледаме тези задачи в следващите глави.

Еднопотребителски режим на работа

В повечето случаи компютърът ви ще работи в многопотребителски режим, в който на потребителите е разрешено да използват системата. Съществува обаче специален *еднопотребителски* режим на работа. В него няма покана за влизане в системата и се използва обикновено от **root.** Може да ви се наложи да използвате този режим, ако възникне някакъв сериозен проблем по време на инсталирането на операционната система. Еднопотребителският режим на работа е важен за някои дейности на системния администратор, например за проверката и поправянето на повредени файлови системи. (Това е доста неприятно и затова трябва да се стремите да не повреждате файловите си системи. Например, винаги трябва да спирате операционната система с командата *shutdown,* преди да изключите захранването на компютъра - виж следващия раздел.)

Компютърът е почти неизползваем в еднопотребителски режим - изпълняват се много малка част от конфигурационните скриптове, не се монтират файловите системи и т.н. Тези ограничения са необходими за

възстановяване на системата след срив - виж раздела "Какво да правим в аварийна ситуация" в Глава 8.

Забележка: Unix е многозадачна система дори в еднопотребителски режим на работа. Можете да пуснете едновременно няколко програми. Сървърите могат да работят във фонов режим, затова могат да работят и специални функции като мрежовата подсистема. Обаче, ако системата ви поддържа повече от един терминал, ще можете да използвате само системната конзола. В еднопотребителски режим няма да работи и системата X Window.

Спиране на системата

За щастие, спирането на работеща Linux система е значително по-просто от зареждането и началната инициализация. Все пак, не трябва просто да натискате бутона Reset. Linux, като всяка друга Unix система, буферира в паметта заявките за четене и писане в твърдия диск. Това означава, че записването на данни в диска се отлага, освен ако не е абсолютно наложително, и в много случаи данните се четат директно от паметта на компютъра, вместо от диска. По този начин се ускорява работата на системата, защото твърдите дискове са изключително бавни в сравнение с централния процесор.

Ако внезапно изключите или рестартирате компютъра си, буферите в паметта няма да бъдат записани на диска и можете да загубите данните си. В повечето системи скриптът ***/etc/rc.d/rc.sysinit*** стартира програмата ***/sbin/update,*** която на всеки пет секунди записва на диска всичкидеза-^ писани буфери (наричат се dirty buffers, тоест "мръсни" буфери; това са тези блокове с данни, които са променени, след като са били прочетени от диска). По този начин няма да се получи сериозна повреда, ако поради някаква причина системата се срине. Все пак, за пълна сигурност, преди изключване или рестартиране на компютъра е необходимо Linux да бъде спрян по "правилния" начин. Така не само дисковите буфери ще бъдат коректно синхронизирани, но и всички процеси ще завършат работата си нормално.

Стандартната команда за спиране или рестартиране на системата е ***shutdown.*** Ако сте влезли като **root,** можете да рестартирате системата след 10 минути като използвате командата:

```
/sbin/shutdown -r +10
```

Опцията -/• означава, че след като завърши процедурата за спирането, системата трябва да бъде рестартирана. Аргументът `+10` задава времето (в минути), което трябва да измине, преди ***shutdown*** да започне процедурата по спирането на системата. На всички активни терминали ще се отпечата предупредително съобщение, че след 10 минути системата ще

бъде спряна. Можете да добавите свое собствено съобщение по следния начин:

```
/sbin/shutdown -r +10 "Rebooting to try new kernel"
```

Можете да зададете абсолютно време за рестартиране. Например, командата:

```
/sbin/shutdown -r 13:00
```

ще рестартира системата в един часа след обяд. Освен това, можете да рестартирате системата веднага, като използвате командата:

```
/sbin/shutdown -r now
```

Разбира се, първо ще се изпълни процесът за спиране на Linux.

Ако вместо *-r* използвате опцията *-h,* системата просто ще бъде спряна, след което ще можете да изключите компютъра си, без да се страхувате, че ще загубите данни. Ако не зададете нито една от тези две опции, след като процедурата за спиране завърши, системата ще премине в еднопотребителски режим.

Както вече споменахме в раздела "Програмата init, файлът inittab и rc-скриптовете", когато натиснете клавишната комбинация Ctrl-Alt-Del, програмата *init* я прехваща и автоматично изпълнява командата *shutdown.* Ако сте свикнали да рестартирате компютъра си по този начин, би било добре да проверите дали във файла */etc/inittab* има ред, който съдържа `ctrlaltdel` в полето **`action.`** Забележка: не трябва никога да изключвате компютъра от копчето за захранване или да използвате бутона Reset за рестартиране на машината. Винаги използвайте командата *shutdown,* освен ако не сте абсолютно сигурни, че системата е увиснала (което се случва много рядко). Едно от предимствата на многозадачните системи е това, че дори ако някоя програма увисне, винаги можете да превключите на друг прозорец или виртуална конзола, за да решите проблема.

Командата *shutdown* предлага немалко други опции. Опцията *-c* се използва за прекъсване на процеса *shutdown,* който изчаква да мине зададеното време, преди да започне процедурата по спиране на системата (разбира се, можете да убиете процеса ръчно с командата *kill,* но.по лесно е да използвате *shutdown -c).* Ако използвате опцията *-k, shutdown* ще отпечата предупредителните съобщения, но няма да спре системата. Ако се интересувате от детайлите, прочетете справочната страница на shutdown(8) `shutdown`.

Файловата система /proc

Unix измина дълъг път в стремежа си да създаде един и същ интерфейс към различните елементи на системата. Както ще научите в следващата глава, в Linux хардуерните устройства се представят като специален тип файл. Освен това, съществува файловата система */proc,* която прави дори нещо повече: тя унифицира достъпа до файловете и процесите.

От гледна точка на потребителя или системния администратор файловата система */proc* изглежда като всяка друга файлова система; можете да се придвижвате в нея с командата *cd,* да прегледате съдържанието на директория с командата *ls* или да разгледате съдържанието на файл с командата *cat.* Но файловете и директориите в тази файлова система не се намират на твърдия диск. Ядрото прехваща заявките към */proc* и генерира в движение директориите и файловете в нея. С други думи, когато разглеждате съдържанието на директория или файл в */proc,* ядрото динамично генерира информацията, която искате да видите.

За да направим това обяснение по-малко абстрактно, ще разгледаме няколко примера. Ето как изглеждат файловете в най-горната директория в йерархията на файловата система */proc:*

```
tigger # ls /proc
1          184        25472      8          8525         kmsg
130        185        25475      82         8526         ksyms
134        186        25497      8484       8593         loadavg
136        187        25498      8485       963          locks
139        2          25499      8488       965          meminfo
143        24924      25500      8489       9654         modules
144        25441      25515      8492       968          mounts
145        25442      25549      8496       97           net
146        25445      25550      8507       99           pci
147        25446      26019      8508       cmdline      scsi
148        25449      26662      8510       cpuinfo      self
151        25451      26663      8511       devices      stat
163        25462      270        8512       dma          sys
168        25463      3          8520       filesystems  uptime
172        25464      4484       8522       interrupts   version
180        25465      4639       8523       ioports
182        25466      55         8524       kcore
```

Във вашата система числата ще бъдат други, но общата организация ще бъде същата. Всяко число е име на директория, която съответства на един от процесите, които работят в системата ви. Нека видим наличната информация за процеса с идентификатор 172:

```
tigger # ls /proc/172
cmdline  environ  fd        mem      stat       status
cwd      exe      maps      root     statm
```

Виждате няколко файла, всеки от които съдържа информация за този процес. Например, файлът *cmdline* съдържа командния ред, с който е

пуснат този процес, *status* дава информация за вътрешното състояние на процеса, а *cwd* е символна връзка към текущата работна директория на процеса (current working directory).

Вероятно хардуерната информация ще ви бъде по-интересна от информацията за процесите. Всичко, което ядрото "знае" за хардуера, се съдържа във файловата система */proc,* макар че понякога е трудно да намерите информацията, която търсите.

Нека започнем с разглеждането на паметта в машината ви. Данните се съдържат във файла */proc/meminfo*\

```
tigger # cat /proc/meminfo
        total:    used:     free:   shared:   buffers: cached:
Mem:  130957312 128684032   2273280  37888000   3198976 20615168
Swap: 133885952 64434176 69451776
MemTotal:    127888 kB
MemFree:       2220 kB
MemShared:    37000 kB
Buffers:       3124 kB
Cached:       20132 kB
SwapTotal:   130748 kB
SwapFree:     67824 kB
```

Ако изпълните командата *free,* ще видите същата информация. Единствената разлика е, че подредбата на числата е леко изменена, *free* не прави нищо повече от прочитането на */proc/meminfo* и пренареждане на изходния резултат.

Повечето програми, които извеждат информация за хардуера в компютъра ви, работят по същия начин. Файловата система */proc* предоставя универсален и лесен начин да получите тази информация. Тя е особено полезна, когато искате да добавите нов хардуер в компютъра си. Например, повечето контролери използват няколко входно/изходни адреса за комуникация с процесора и операционната система. Ако конфигурирате две устройства така, че да използват едни и същи входно/изходни адреси, ще имате проблеми. Можете да избегнете подобни неприятности, като проверите кои входно/изходни адреси се използват от драйверите на ядрото:

```
tigger # more /proc/ioports
0000-001f   dma
0020-003f   pic1
0040-005f   timer
0060-006f   keyboard
0080-009f   dma page reg
00a0-00bf   pic2
00c0-00df   dma2
00f0-00ff   npu
01f0-01f7   ide0
0220-022f   soundblaster
02e8-02ef   serial (auto)
0388-03b8   OPL3/OPL2
```

```
03cO-03df   vga +
03f0-03f5   floppy
03f6-03f6   ideO
03f7-03f7   floppy DIR
03f8-03ff   serial (auto)
0530-0533   WSS config
0534-0537   MSS audio codec
eOOO-eOЬe   aic7xxx
e400-e41f   eth0
```

Сега можете да потърсите свободни входно/изходни адреси. Разбира се, ядрото показва входно/изходните адреси само на откритите и разпознати контролери, но в една правилно конфигурирана система ще бъдат разпознати всички устройства.

От */proc* можете да получите и друга информация, необходима за конфигуриране на новия хардуер - във файла */proc/interrupts* са изброени използваните IRQ линии, а в */proc/dma* са изброени използваните канали за директен достъп до паметта (DMA).

Управление на потребителски account

Важно е да разбирате как се управлява потребителски account, дори ако вие сте единствения човек, който използва вашата система. Още по-важно е да разбирате как става това, ако на системата ви работят много потребители.

В Unix описанието на потребител (тоест потребителския account) се използва за много цели. Очевидно, като използва описанията на потребителите, системата може да различи хората, които работят на нея. Това се прави с цел идентификация на потребителите и гарантиране на сигурността на системата. Всеки потребител има собствен account, в който е записано името, с което се представя в системата, и паролата, която използва. В раздела "Собственост и права за достъп до файл" в Глава 4 обяснихме как потребителите могат да задават права за достъп до файловете си, така че да ограничат или разрешат достъпа до тях на други потребители. Всеки файл в системата се "притежава" от конкретен потребител, който може да зададе правата за достъп до този файл. Описанията на потребителите се използват за удостоверяване на самоличността на потребителя, който иска да влезе в системата; само хората с account могат да използват машината. Освен това, потребителските описания се използват в системните log-файлове, за отбелязване на подателя на електронната поща и т.н.

Глава 4

В системата съществуват няколко account-а с административни функции. Както видяхме, системният администратор използва account-а root, за да извърши необходимите действия по поддръжката на системата. Обикновено root не се използва за нормална работа. Достъпът до такива

account-и се осъществява с командата *su,* която ви позволява да използвате друг account, след като сте влезли в системата със собствения си account.

Някои account-и не са предназначени за хората, работещи в системата. Обикновено те се използват от някои системни демони, които трябва да имат достъп до файловете в системата чрез специфичен идентификатор, различен от идентификатора на root или друг потребител в системата. Например, ако конфигурирате Linux да получава Usenet съобщения от друга система, news-демона трябва да запише тези съобщения в директория, от която може да чете всеки потребител, но в която може да пише само един "потребител" - самият демон. Никой човек не се асоциира с account-а **news** - той е за "въображаем" потребител и съществува само за news-демона.

За изпълнимите файлове може да се зададе специално право за достъп, което се нарича *setuid* (set user identifier). Когато е зададено това право, програмата получава правата, които има собственикът на файла. Например, ако файлът, в който се намира news-демонът, е собственост на потребителя **news** и е зададено правото *setuid,* демонът ще има правата на потребителя **news.** Потребителят **news** би могъл да има прав^ да пише в директорията, в която се съхраняват Usenet съобщенията, а останалите потребители - само да четат файловете в нея. Това е един от начините, по които работи системата за сигурност. Така потребителите ще могат да четат съобщенията в Usenet, но никой не може просто "да си играе" с файловете в тази директория.

Една от задачите ви като системен администратор е да създавате и управлявате account-и на потребителите (истински и въображаеми) в своята система. В повечето случаи това не е тежка задача, но е важно да знаете какво трябва да извършите.

Файлът /etc/passwd

За всеки account в системата съществува запис във файла */etc/passwd.* Този файл съдържа записи (по един ред за всеки потребител), които задават различните атрибути на всеки account, например истинското име на потребителя, името, с което се представя в системата и т.н.

Всеки ред от този файл има следния формат:

```
username:password:uid:gid:gecos:homedir:shell
```

Следва описание на полетата в записа:

username

Уникална последователност от символи, с която се идентифицира този account. Ако account-ът се използва от човек, това е името, с

което той влиза в системата. В повечето системи максималната дължина е осем символа. Примери - **larry, kristen.**

password

Паролата на потребителя в кодиран вид. Това поле се променя с програмата *passwd* (с нея се задава паролата на потребителя); кодирането е такова, че не съществува обратна функция за декодиране на паролата, затова е много трудно (ако не и невъзможно) от това поле да се разбере каква е паролата. Стойността на това поле не се задава ръчно; програмата *passwd* го попълва автоматично. Забележка: ако първият символ на полето е *, този account е "забранен" - т.е. не е позволено потребителя да влезе в системата (виж раздел "Създаване на нов account" по-долу в тази глава).

uid

Идентификатор на потребителя (user ID), уникално цяло число, което се използва от системата за означаване на този account. Системата използва *uid,* когато работи с правата за достъп до файловете и процесите, защото е по-лесно да се работи с едно цяло число (тоест *uid),* вместо с последователност от символи *(username). uid* и *username* задават един и същ account; за системата е по-важЯа стойността на *uid,* а за потребителя е по-лесно да използва *username.*

gid

Идентификатор на групата (group ID), цяло число, което задава групата, на която принадлежи потребителят. Групите са описани във файла */etc/group.* Потребителят може да принадлежи на няколо групи, но тук се задава само една от тях - подразбиращата се група (виж раздела "Файлът /etc/group").

gecos

Друга информация за потребителя, например истинското име на потребителя, адрес, телефон и т.н. Програми като *mail* и *finger* използват тази информация за идентифициране на потребителите на системата; ще обсъдим това по-късно. Между другото, *gecos* е историческо име, което датира от 70-те години на XX век и е съкращение на *General Electric Comprehensive Operating System* (пълна операционна система на General Electric). GECOS няма нищо общо с Unix, освен полето *gecos,* което е добавено в */etc/passwd* за съвместимост на Unix с някои от услугите на GECOS.

homedir

Личната директория на потребителя (виж по-долу). Когато влезе в системата, потребителят започва да работи в тази директория.

shell

Името на програмата, която ще се стартира, когато потребителят влезе в системата; в повечето случаи това е пълното име на командния интерпретатор, който потребителят използва, например */bin/bash* или */bin/tcsh*.

Задължителни са само полетата ***username, uid, gid*** и ***homedir.*** В повечето потребителски account-и всички полета са запълнени, но "въображаемите" или административните account-и обикновено използват само някои от полетата.

Ето два примерни реда от */etc/passwd:*

```
root:ZxPsI9ZjiVd9Y:0:0:The root of all evil:/root:/bin/bash
aclark:BjDf5hBysDsii:104:50:Anna Clark:/home/aclark:/bin/bash
```

Първият ред е за account-а **root.** Най-важното е, че ***uid*** в този ред е нула.

Точно затова **root** е **root** - системата знае, че ***uid*** 0 е специален и за него не се прилагат стандартните ограничения. Полето ***gid*** също е нула, което е по-скоро конвенция. Много от файловете в системата са собственост на потребителя **root** и групата **root,** които имат съответно ***uid*** и ***gid*** равни на нула. След малко ще обсъдим групите по-подробно.^

В много системи личната директория на **root** е */root* или просто /. Това обикновено не е важно, защото най-често ще използвате командата *su* от собствения си account. Освен това, обикновено **root** използва вариант на Bourne shell (в този случай *Mn/bash),* макар че ако искате, можете да използвате вариант на C shell (командните интерпретатори са разгледани в раздела "Командни интерпретатори" в Глава 4). Бъдете внимателни: командните интерпретатори, които са съвместими с Bourne-shell, използват различен синтаксис от тези, които са съвместими със C shell. Затова промяната на командния интерпретатор на потребителя **root** може да причини неприятности.

Ш

Глава4

Вторият ред е за човек с потребителското име **aclark.** В^този случай полето и2с7съдържа стойност 104. Технически това поле може да съдържа произволно уникално цяло число; в много системи номерът на потребителските account-и е число по-голямо от 100, защото числата до 100 са запазени за административни цели. Полето ***gid*** съдържа числото 50, което означава, че **aclark** принадлежи на групата, която има идентификатор 50 във файла */etc/group* (групите са описани по-късно в тази глава).

Личните директории се намират обикновено в директорията */home* и имената им съвпадат с потребителските имена на техните собственици. Тази конвенция помага в повечето случаи, когато е необходимо да се открие личната директория на конкретен потребител, но технически можете да поставите личната си директория на произволно място. Все

пак би трябвало да спазвате конвенцията, която се използва във вашата система.

Забележка: не е необходимо системният администратор директно да променя файла */etc/passwd.* Съществуват множество програми, които ще ви помогнат да създавате и управлявате потребителските account-и (виж раздела "Създаване на нов account").

Скрити пароли

В известен смисъл съществува риск за сигурността, ако всеки потребител, който има достъп до системата, може да прочете кодираните пароли в */etc/passwd.* Съществуват специални програми, които могат да изчислят кодиращата функция за огромно количество потенциални пароли и да проверят дали кодираната версия на потенциалната парола съвпада със стойността в съответното поле.

За да се премахне този риск, са измислени скритите пароли (shadow passwords). Когато използвате скрити пароли, полето *password* във файла */etc/passwd* вместо кодираната парола съдържа само символа x или *. Самата кодирана парола се намира във файла */etc/shadow,* който има редове, подобни на тези в */etc/passwd,* но полето *password* съдържа същинската кодирана парола. Файлът */etc/shadow* може да се чете само от root, затова обикновените потребители не могат да получат достъп до кодираните пароли. Полетата в */etc/shadow,* с изключение на *username* и *password,* обикновено са празни или съдържат фалшива информация.

За да използвате скрити пароли, трябва да имате специални версии на програмите, които четат или променят потребителските account-и, например *passwd* и *login.* Днес повечето дистрибуции могат да използват скрити пароли, така че едва ли ще имате такива проблеми.

Съществуват две програми, с които можете да превърнете "обикновените" пароли в скрити и обратно. Програмата *pwconv* четг файла */etc/passwd,* намира записите, които все още не се намират в */etc/shadow,* генерира съответните редове и ги добавя към вече съществуващите записи в */etc/shadow.*

Потребителите на Debian трябва вместо това да използват командата "shadowconfig on", за да разрешат употребата на скрити пароли.

Програмата *pwunconv* се използва сравнително рядко, защото намалява сигурността на системата. Тя извършва обратното действие на *pwconv,* като взима кодираните пароли от */etc/shadow* и ги връща в */etc/passwd.*

PAM и други методи за удостоверяване на самоличността

Грешите, ако мислите, че е достатъчно да има два начина за удостоверяване на самоличността на потребителя - */etc/passwd* и */etc/shadow*. Съществуват няколко други метода за удостоверяване на самоличността със странни имена, например Kerberos authentication (по името на кучето, което пази входа към подземното царство в гръцката митология - Цербер). Смятаме, че скритите пароли почти винаги гарантират достатъчна степен на сигурност, но всичко зависи от степента на сигурност, от която се нуждаете и от това колко сте параноичен.

Проблемът при тези методи е, че не можете просто да превключите от единия на другия метод. Системата трябва да се настрои така, че да използва програми като *login* и *passwd,* които са част от софтуера, който реализира съответния метод. За да се реши този проблем, са създадени РЛМ-системите (съкращение на Pluggable Authentication Method - сменяеми методи за удостоверяване на самоличността). Ако използвате РЛМ-софтуер, можете да промените метода за удостоверяване^а самоличността, просто като преконфигурирате PAM. Системата ви автоматично ще стартира кода, който трябва да изпълни необходимите процедури за удостоверяване на самоличността (този код се намира в динамично свързани библиотеки).

Настройката и използването на PAM е извън обсега на тази книга, но можете да получите необходимата информация за това от сайта *http://www.de.kernel.org/pub/limix/libs/pam/*.

Файлът /etc/group

Групите са удобен начин за логическа организация на потребителите в системата и позволяват на хората в една група съвместно да използват определени файлове. Всеки файл в системата се асоциира едновременно с един потребител и с една група. С командата *ls -1* можете да видите собственика и групата на даден файл:

```
rutabaga% ls -1 boiler.tex
-rwxrw-r--   1 mdw    megabozo  10316 Oct  6 20:19 boiler.tex
rutabaga%
```

Файлът е собственост на потребителя **mdw** и е асоцииран с групата **megabozo.** От правата за достъп виждаме, че **mdw** може да чете, пише и изпълнява този файл; всеки член на групата **megabozo** може да чете и пише във файла; всички останали потребители могат само да го четат.

От това не следва, че **mdw** е член на групата **megabozo;** просто файлът
може да се чете и пише от всеки член на групата **megabozo** (която може да включва, но може и да не включва **mdw).**

По този начин файловете могат да се използват съвместно от групи потребители и правата за достъп могат да се задават отделно за собственика на файла, за групата, с която е асоцииран файла и за всички останали потребители (виж раздела "Собственост и права за достъп до файл" в Глава 4).

Глава 4

Всеки потребител е член поне на една група, която се задава в полето
gid във файла */etc/passwd.* Освен на тази група, потребителят може да е член на още много групи. Файлът */etc/group* съдържа едноредов запис за всяка група в системата, и има формат, подобен на формата на */etc/passwd.*

Форматът на */etc/group* е следния:

```
groupname:password:gid:members
```

Тук *groupname* е символен низ, който идентифицира групата; това е името на групата, което се отпечатва, когато използвате команди като *ls -l.*

password е незадължителна парола, която е асоциирана с групата и позволява на потребители, които не са членове на тази група, да я използват с командата *newgrp* (виж по-долу).

gid е идентификатора на групата (group ID), който се използва от системата за означаване на групата; това е числото, което се използва в щь лето *gid* във файла */etc/passwd,* за да се зададе подразбиращата се група на потребител.

members е списък с потребителски имена (за разделител се използва запетайка; не трябва да има интервали или табулации), в който са включени всички потребители, които са членове на тази група, но имат различна стойност на полето *gid* във файла */etc/passwd.* Тоест, този списък не съдържа потребителите, за които тази група е зададена като "подразбираща" се в */etc/passwd,* а само тези, за които тя е допълнителна група.

Например, файлът */etc/group* може да съдържа следните редове:

```
root:*:0:
bin:*:1:root,daemon
users:*:50:
bozo:*:51:linus,mdw
megabozo:*:52:kibo
```

Първите редове съдържат групите **root** и **bin,** които са административни групи и се използват като "въображаеми" потребители в системата. Много файлове са асоциирани с групи като **root** и **bin.** Другите групи в

този пример са за обикновени потребители. Както идентификаторите на потребителите, така и идентификаторите на групите за обикновени потребители обикновено имат стойност, която е по-голяма от 50 или 100.

Полето *password* (парола) във файла */etc/group* се използва рядко. С помощта на програмата *newgrp* потребители, които не са членове на дадена група, могат да използват идентификатора й, ако знаят паролата. Например, ако използвате командата:

```
rutabaga% newgrp bozo
Password: паролата за групата bozo
rutabaga%
```

ще се стартира нов команден интерпретатор с идентификатора на групата bozo. Ако полето *password* е празно или съдържа символа *, ще получите грешката permission denied (отказан достъп), ако се опитате да използвате командата *newgrp* за тази група.

Полето *password* във файла */etc/group* рядко има стойност и практически не е необходимо. (Всъщност, повечето системи не предлагащ програма, с която да зададете парола за това поле; бихте могли да използвате *passwd,* за да зададете парола на фиктивен потребител със същото име като това на групата, след което да копирате кодираната парола от */etc/passwd* в */etc/group.)* Вместо да използвате това поле, можете да направите потребителя член на много групи, просто като добавите името му в полето *members* на всяка от тях. В горния пример потребителите linus и mdw са членове на групата bozo, както и на групата, която е^ададена в съответното поле на файла */etc/passwd.* Ако искаме да добавим Iinus и към групата megabozo, трябва да променим последния ред в примера по следния начин:

```
megabozo:*:52:kibo,linus
```

Командата *groups* ви казва на кои групи принадлежите, например:

```
rutabaga% groups
users bozo
```

Ако зададете като аргумент списък с потребителски имена, *groups* ще отпечата групите, на които принадлежи всеки потребител.

Когато влизате в системата, автоматично получавате идентификатора на групата, който е зададен в записа за вашия account във файла */etc/passwd,* както и идентификаторите на групите, за които сте описани като член във файла */etc/group.* Това означава, че за всеки файл, който е асоцииран с някоя от групите, на които сте член, получавате правата за достъп на групата (зададени с *chmod g+...* ; ако сте собственик на файла, за вас се прилагат правата за достъп на собственика).

Сега, след като знаете за какво се използват групите, как трябва да разпределите потребителите в групи? Това е преди всичко въпрос на стил и зависи от това как се използва системата ви. За системи, които имат малко потребители, най-лесно е да се използва една единствена група (например **users),** в която ще членуват всички потребители. Забележка: не трябва да се променят системните групи (това са групите, които се създават в */etc/group* при инсталирането на системата), защото от тях зависи работата на различни програми и демони в системата ви.

Ако системата ви се използва от много потребители, можете да ги организирате в групи по много начини. Например, един университет може да използва отделни групи за студентите, преподавателите и останалия персонал - съответно **students, faculty** и **staff**/Една софтуерна компания може да има група за всеки от екипите, които работят по различни проекти. В други системи, всеки потребител е член на група, която има неговото име (така да се каже, всяка жаба си знае гьола), и потребителите могат да използват съвместно файлове с произволен друг потребител в системата (при това, само с него).

Все пак, добавянето на потребител към друга група обикновено изисква намеса на системния администратор (трябва да се редактира файлът */etc/group;* Debian предлага програмата *gpasswd).* Наистина, всичко зависи от вас.

Често групите се използват за определяне на потребителите, които могат да използват определено хардуерно устройство. Нека, например, имате скенер и той се използва чрез специалния файл */dev/scanner.* Ако не искате всеки да има достъп до скенера, можете да извършите следната процедура: да създадете специална група, която се казва **scanner;** да асоциирате */dev/scanner* с нея; да зададете право за четене от този файл за групата **scanner** и да премахнете правото за четене за останалите потребители; и накрая, да добавите всеки, който ще може да използва скенера към тази група, като редактирате файла */etc/group.*

Създаване на нов account

Създаването на потребителски account се извършва на няколко стъпки - добавяне на запис в */etc/passwd,* създаване на лична директория за новия потребител и накрая - създаване на конфигурационни файлове (като *.bashrc)* в тази директория.

•

Обикновено не се налага да извършвате тези стъпки ръчно; почти всички Linux системи съдържат програмата *adduser,* която може да направи това.

Ще покажем как можете да използвате *adduser* като root. След като стартирате програмата, трябва просто да въвеждате необходимата информация, когато *adduser* поиска параметрите на новия потребител; когато искате да използвате подразбиращата се стойност, просто натиснете Enter:

```
Adding a new user. The username should not exceed 8 characters in
length, or you may run into problems later.
 (Добавяне на нов потребител. Потребителското име не трябва да е по-
дълго от 8 символа, защото по-късно можете да имате проблеми.)
Enter login name for new account (^C to quit): norbert
 (Въведете името, с което ще се влиза в системата (^C за прекъсване):
norbert)
Editing information for new user [norbert]
 (Редактиране на информацията за новия потребител [norbert])
Full Name (пълно име): Norbert Ebersol ^
GID [100]: 117
Checking for an available UID after 500
First unused uid is 501
 (Проверка за свободен UID след 500
Първият неизползван uid е 501)
UID [501]: (натиснете Enter)
Home Directory (лична директория) [/home/norbert]: натиснете Enter
Shell (команден интерпретатор) [/bin/bash]: (натиснете Enter)
Password [norbert]: (паролата на norbert)
Information for new user [norbert]:
Home directory: [/home/norbert] Shell: [/bin/bash]
Password: [(паролата на norbert)] uid: [501] gid: [51]
Is this correct? [y/N]: y
Вярно ли е това? [y/N]: y
Adding login [norbert] and making directory [/home/norbert]
```

*

Забележка: Някои Linux системи, например Red Hat и SuSE, използват други програми за създаване и изтриване на account. Ако примерите в този раздел не работят във вашата система, проверете документацията за дистрибуцията, която използвате. (Red Hat използва програмата *control-panel* за управление на acountите, SuSE използва *YaST* за същата цел; в Debian скриптът "adduser" автоматично създава account, като използва конфигурационния файл */etc/adduser.conf).* Освен това, съществуват графични програми за управляване на потребителските account-и, например *kuser* от KDE (виж раздела "Работната среда KDE" в Глава 11).

```
Adding the files from the /etc/skel directory:
 ./.emacs -> /home/norbert/./.emacs
 ./.kermrc -> /home/norbert/./.kermrc
 ./.bashrc -> /home/norbert/./.bashrc
```

Тук не би трябвало да има изненади; достатъчно е да въведете необходимата информация или да изберете подразбиращите се стойности. Забележка: *adduser* използва 100 като подразбиращ се идентификатор на група и търси първия неизползван потребителски идентификатор след 500 (това е минималната стойност в SuSE и Red Hat, Debian използва 1000). При използване на подразбиращите се стойности би трябвало да не възникнат проблеми; в този пример използваме идентификатор на група 117 и подразбиращия се идентификатор на потребител - 501.

След като се създаде новият account, файловете от директорията */etc/skel* се копират в личната директория на потребителя, */etc/skel* съдържа "примерни" (skeleton) файлове - това са подразбиращите се конфигурационни файлове за новия потребител (например *.emacs* и *.bashrc).* Можете свободно да добавяте избрани от вас файлове в */etc/skel,* ако смятате, че новите потребители в системата ви трябва да ги имат.

След като завърши работата на *adduser,* новият account е готов за използване и **norbert** може да влезе в системата, като използва паролата, която е зададена с *adduser.* За да се гарантира сигурността на системата, веднага след влизането си в системата, новите потребители трябва да сменят паролите си с командата *passwd.*

root може да зададе паролата за произволен потребител в системата. Например командата:

```
passwd norbert
```

ви дава възможност да зададете новата парола на **norbert,** без да е необходимо да въвеждате старата парола. Забележка: за да смените паролата на **root,** трябва да я знаете. Ако забравите паролата на **root,** можете да заредите Linux от дискета и да изтриете съдържанието на полето *password* на реда за **root** във файла */etc/passwd* (виж раздела "Какво да правим в аварийна ситуация" в Глава 8).

Някои системи вместо *adduser* съдържат програмата *useradd,* която създава нов потребител, като необходимата информация се задава във вид на аргументи от командния ред. Ако се налага да използвате *useradd,* прочетете справочната страница, в която са описани необходимите параметри.