ИНСТАЛЛЯЦИЯ LINUX И ПЕРВЫЕ ШАГИ

М. Уэлш

Contents

Alexander,

Thanks for the note about the translation! This is very good work, and I appreciate your work. Hopefully you will be able to explore publishing the translation on paper.

Regards! M. Welsh
24 April 1996

ПРЕДИСЛОВИЕ ПЕРЕВОДЧИКА

Госкомитет СССР по науке и технике в справедливой борьбе за чистоту русского технического языка, настоятельно рекомендовал (а может и требовал) в одном из своих циркуляров использовать "многопрограммирование" вместо "мультипрограммирование" "сопряжение" вместо "интерфейс" и т.п. А сколько боролись со словом "компьютер", вместо которого был достаточный выбор "русских слов": хочешь - ЭВМ, хочешь - ЦВМ, хочешь - ЭЦВМ.

Граница между своими и чужими словами в вычислительной технике живая и подвижная. Никто сегодня не требует русского названия для "драйвера", как и английского для "ядра".

Переводчик в большей степени руководствовался находящимся в обращении "техническим жаргоном", стараясь при этом не потакать экстремистам, говорящим "рутина", "хадвер", "сорс" и "фейсом об тейбл".

При переводе сделаны очень небольшие сокращения, в основном в начале первой главы, носящей обзорный характер и наиболее подверженной разрушительному действию времени.

Книга будет полезна начинающим пользователям Linux, при этом даже изложение основ UNIX ощутимо отличается от типового для такого рода литературы по операционным системам этого семейства.

Автор, чтобы подсластить сухую техническую информацию и не отпугнуть возникающими трудностями новичка, временами допускает некоторую игривость слога, в чем переводчик охотно следовал за ним.

Перевод выполнен с любезного разрешения автора. А в благодарностях нельзя обойти *Алексея Костарева* (`mailto:kaf@nevod.perm.su`), выполнившего большой об'ем научного и технического редактирования. Без разнообразных вдохновляющих воздействий соседей из Екатеринбурга (*Павла Шабашова* (`mailto:pavel@vltava.mplik.ru`), *Евгения Шахтарина* (`mailto:eugene@mplik.ru`), *Сергея Червинского* (`mailto:serge@abacus-ekt.mplik.ru`), *И.А. Бокова* (`mailto:igor@abacus-ekt.mplik.ru`)) этот перевод был бы выполнен (если выполнен) существенно позже.

Заявляя авторские права на перевод ИНСТАЛЛЯЦИЯ LINUX И ПЕРВЫЕ ШАГИ следует очертить их границы:

1. Перевод может свободно распространяться на основе лицензии GPL (см. Приложение E). При полном или частичном воспроизведении перевода обязательно указание авторских прав на перевод.
2. *Переводчику* (`mailto:sae@mobil.perm.su`) хотелось бы знать о любых планах публикации или распространения данного перевода на коммерческой основе. При этом не должны нарушаться лицензионные ограничения, наложенные на оригинал книги и включенные в перевод книги: *Matt Welsh "Linux Installation and Getting Started"*.

Перевод книги в форматах HTML, SGML, TXT можно найти, например, на узле *НЕВОД* (`http://nevod.perm.su/service/linux/doc/lig/`).
А. Соловьев (`mailto:sae@mobil.perm.su`)
10.04.1996

Версия 2.2.2, 11 февраля 1995.

Эта книга одновременно и руководство по инсталляции (установке), и руководство для новичка в Linux. Это означает, что книга может быть интересна и тому, для кого за словом UNIX стоит что-то новое, и тем, кто уже имеет определенный опыт. Здесь есть информация о том, как достать Linux, как его инсталлировать, учебные материалы для новичков в UNIX и введение в администрирование системой. Это подразумевает, что книга рассматривает все с достаточно общих позиций, позволяющих ее использовать при работе с различными дистрибутивами Linux.

Эта книга свободно распространяется; вы можете ее копировать и сами распространять при выполнении определенных условий. Пожалуйста, ознакомтесь с авторскими правами и условиями распространения, приведенными далее.

ПРЕДИСЛОВИЕ

“Вы в лабиринте узких извивающихся ходов, да еще и похожих друг на друга”. Перед вами маячит одна из наиболее сложных и до жути страшных систем, из когда-либо написанных живыми существами. Linux - это операционная система, свободнораспространяемая иликороче – *free*, относящаяся к семейству "клону" UNIX для персональных компьютеров, созданная сумбурной командой UNIX-экспертов, хакеров и случайно прибившихся к этой честной компании еще более подозрительных личностей.

Хотя система невольно отражает эту тяжелую наследственность, и хотя процесс создания Linux выглядел как дезорганизованные усилия добровольцев, система получилась на удивление мощной, быстрой и бесплатной. Это блестящее решение в классе 32-разрядных операционных систем.

Лично мое знакомство с Linux началось несколько лет назад, когда я задумался, как инсталлировать единственный "дистрибутив" Linux, существовавший в то время - пара дискет, подготовленных H.J. Lu. Я сбросил кучу файлов и стал читать, страница за страницей, неряшливо подготовленную инструкцию по инсталляции. Каким-то образом мне удалось инсталлировать эту базовую систему и заставить все вместе работать. Это было задолго до того, как стало возможно приобрести Linux у дистрибуторов всего мира на CD-ROM. Еще до того, когда сам Linux научился обращаться к CD-ROM. Это было до XFree86, до Emacs, до организации коммерческого сопровождения, до того, как Linux стал действительным соперником MS-DOS, Microsoft Windows и OS/2 на рынке операционных систем персональных компьютеров.

Вы держите в своих собственных руках карту и путеводитель в мире Linux. Я надеюсь, что эта книга поможет вам в приятном путешествии по тому, что я считаю быстрейшей и самой мощной операционной системой для персональных компьютеров. Установить свой собственный Linux - это может быть само по себе большим удовольствием, так что берите чашку кофе, садитесь и начинайте читать. И для меня тоже чашечку, раз уж вы этим занялись. Я ведь уже давно тут, в Linux, живу.

АУДИТОРИЯ

Эта книга для всех пользователей персональных компьютеров, которые хотят установить и использовать Linux. Мы предполагаем, что вы имеете базовые представления о персональных компьютерах и операционных системах, вроде MS-DOS. Предварительных знаний по Linux или UNIX не предполагается.

Несмотря на это, мы настоятельно рекомендуем потратиться на одну из многочисленных хороших книг по UNIX. Некоторые из них перечислены в Приложении A.

СТРУКТУРА КНИГИ

Эта книга состоит из следующих глав.

Глава 1. Введение в LINUX.

Дает общее представление о том, что представляет из себя Linux, что он может для вас сделать, что требуется для его эксплуатации на вашей системе. В главе также содержатся полезные советы и подсказки, как помочь самому себе и избежать стрессов.

Глава 2. Приобретение и инсталляция Linux.

Об'ясняет, как достать Linux software и как его инсталлировать, начиная с переразбиения диска, создания файловых систем и загрузки программ. Глава содержит достаточно общие инструкции, применимые ко всевозможным дистрибутивам Linux, что в сочетании с документацией, поставляемой с конкретной версией, позволит найти ответы на все вопросы.

Глава 3. Знакомство с Linux.

Это исчерпывающее введение в использование Linux для новичков в UNIX. Если вы имеете предварительный опыт работы с UNIX, большая часть этого материала вам знакома.

Глава 4. Администрирование.

Обсуждает много важных концепций, связанных с администрированием Linux. Материал может быть также интересен администраторам UNIX-систем, которые хотели бы узнать об особенностиях эксплуатации Linux.

Глава 5. Дополнительные возможности.

Представляет читателю ряд дополнительных возможностей, поддерживаемых системой Linux, таких как X Window System и TCP/IP. Включено полное руководство по настройке XFree86-3.1.

ПРИЛОЖЕНИЕ A. Источники информации по Linux.

Содержит перечень других источников информации по Linux, включая группы телеконференций, списки рассылки, документы, доступные по он-лайн, и книги.

ПРИЛОЖЕНИЕ B. Список поставщиков Linux.

Содержит небольшой список поставщиков, предлагающих Ос Linux и и различный сервис.

ПРИЛОЖЕНИЕ C. Учебные материалы по FTP и список серверов.

Содержит учебные материалы по скачиванию файлов из Internet по FTP. Это приложение также включает список архивных серверов FTP, содержащих программное обеспечение Linux.

ПРИЛОЖЕНИЕ D. Список BBS для Linux .

Содержит список BBS, содержащих программное обеспечение Linux. Поскольку большинство пользователей Linux не имеют выхода в Internet, важно то, что информация с BBS стала доступной.

ПРИЛОЖЕНИЕ E. Генеральная Общедоступная Лицензия GNU.

Содержит копию GNU GPL, лицензионные соглашения, по которым распространяется Linux. Очень важно, чтобы пользователи Linux понимали GPL; много недоразумений в связи с требованиями GPL возникало в последние месяцы.

ПРИЛОЖЕНИЕ F. The GNU General Public License.

Оригинальный текст лицензии на английском языке.

БЛАГОДАРНОСТИ

Эта книга создавалась долго и многие люди причастны к ее выходу в свет. Особенно я хотел бы поблагодарить Larry Greenfield и Karl Fogel за их работу над первой версией третьей главы, и Lars Wirzenius, за его работу над четвертой главой. Я благодарен Michael K. Johnson за его помощь с проектом LDP и использованием LaTeX при оформлении книги, а также Ed Chi, который прислал мне подготовленный для издания вариант книги.

Спасибо Melinda A. McBride из SSC, Inc., которая отлично подготовила индекс для глав 3, 4 и 5. Мне также приятно поблагодарить Andy Oram, Lar Kaufman и Bill Hahn из фирмы O'Reilly and Associates за их помощь в Проекте Документирования Linux (Linux Documentation Project или LDP).

Я благодарен Linux Systems Labs, Morse Telecommunications, Yggdrasil Computing и другим компаниям за их поддержку LDP путем продажи книг и других работ.

Большое спасибо многим активистам, включая (не обращайте внимания на порядок перечисления) Linus Torvalds, Donald Becker, Alan Cox, Remy Card, Ted T'so, H.J. Lu, Ross Biro, Drew Eckhardt, Ed Carp, Eric Youngdale, Fred van Kempen, Steven Tweedie и многих других, за то, что они потратили много времени и энергии на этот проект, и без кого не было бы того, о чем написано в этой книжке.

Особая благодарность мириадам читателей, которые присылали свои полезные советы и замечания. Здесь еще очень многих следовало бы упомянуть. Зачем нужна программа проверки правописания, если у вас такая большая аудитория?
Matt Welsh
13 января 1994

ДОВЕРИЕ И ЗАКОННОСТЬ

The Linux Documentation Project - это не жестко определенная команда писателей, корректоров и редакторов, которые работают над определенными руководствами по Linux. Главный координатор проекта Matt Welsh, которому помогают Lars Wirzenius и Michael K. Johnson.

Это руководство - одно из множества распространяемых в рамках Linux Documentation

Project, включая Linux User's Guide, System Administrator's Guide и Kernel Hacker's Guide. Эти руководства доступны в форматах LaTeX и Postscript через anonymous FTP-сервер `sunsite.unc.edu` из каталога `/pub/Linux/docs/LDP`.

Мы призываем всех, кто любит писать или редактировать, присоединиться к нам с целью улучшения документации по Linux. Если у вас есть выход на электронную почту Internet, вы можете присоединиться к каналу DOC - списку рассылки Linux-Activists, послав письмо по адресу:

```
linux-activists-request@niksula.hut.fi
```

с первой строкой послания, имеющей вид

```
X-Mn-Admin:  join DOC
```

Не стесняйтесь выходить на контакт с автором и координатором этого руководства, если у вас есть, вопросы, почтовые открытки или деньги. С Matt Welsh можно связаться по электронной почте Internet по адресу mdw@sunsite.unc.edu, а в реальной жизни по адресу

205 Gray Street
Wilson, N.C. 27896
U.S.A.

UNIX*TM* - торговая марка X/Open. Linux - не торговая марка и не имеет отношения к UNIX*TM* или X/Open. The X Window System *TM* - торговая марка Massachusetts Institute of Technology. MS-DOS *TM* и Microsoft Windows*TM* - торговые марки Microsoft, Inc.

ИНСТАЛЛЯЦИЯ LINUX И ПЕРВЫЕ ШАГИ может тиражироваться и распространяться целиком или по частям при выполнении следующих условий:

1. Приведенный выше copyright и эти условия должны приводиться целиком во всех частичных и полных копиях.
2. Любой перевод или вторичная работа, базирующаяся на ИНСТАЛЛЯЦИЯ LINUX И ПЕРВЫЕ ШАГИ, должна быть одобрена автором в письменной форме до начала распространения.

3. Если вы распространяете ИНСТАЛЛЯЦИЯ LINUX И ПЕРВЫЕ ШАГИ по частям, то должны включаться инструкции, как можно получить полную версию этого руководства и предоставлены средства получения такой копии.
4. Небольшие части могут воспроизводиться в других работах как иллюстрации в обзорах или цитаты без этих разрешительных указаний, при условии корректного цитирования.
5. Генеральная лицензия GNU (The General Public License) которая приводится ниже может быть приведена при условиях, указанных в ней.
6. Некоторые разделы этого документа находятся под отдельным copyright. Когда эти разделы попадают под различные copyright, то указываются все copyright. Если вы распространяете ИНСТАЛЛЯЦИЯ LINUX И ПЕРВЫЕ ШАГИ по частям и данная часть попадает под несколько более общих copyright, то указываются условия copyright этой части.

Исключения из этих правил могут быть предоставлены для академических целей: напишите Matt Welsh по ранее указанному адресу или по электронной почте на mdw@sunsite.unc.edu и спросите разрешения. Эти ограничения вводятся для того, чтобы защитить авторские права, но не ограничить вас как учителей и обучаемых. Автор всячески поощряет дистрибуторов Linux использовать книгу как руководство по инсталляции и первому знакомству с Linux. В соответствии с приведенным выше copyright вы можете свободно печатать и распространять эту книгу бесплатно или получать прибыль. При этом вы можете захотеть включить небольшое “дополнение по инсталляции” для вашей версии.

Автор хотел бы знать о любых планах коммерческого издания и распространения книги. Тогда мы можем информировать вас о всех последних изменениях. И окажись новая версия книги букально за ближайшим поворотом, может вы решите отложить свое издание до появления новой версии книги.

Если вы распространяете книгу на коммерческой основе, то вознаграждения, проценты и/или экземпляры изданной книги также принимаются автором с благодарностью.

Ваши взносы такого рода свидетельствуют о вашей поддержке бесплатных программных продуктов и проекта LDP.

Весь исходный текст книги *Linux Installation and Getting Started* помещен под Генеральной лицензией GNU. Смотрите в ПРИЛОЖЕНИИ E GNU “GPL.”

УСЛОВНЫЕ ОБОЗНАЧЕНИЯ

В книге использованы следующие обозначения:

Жирным фонтом

выделяются слова, обозначающие **новые концепции**, **ПРЕДУПРЕЖДЕНИЯ** или **ключевые слова** языка.

Курсивом

выделяются *ударения в тексте*, *названия документов* и пр.

`Фиксированным фонтом`

представлены `экранные формы`. Также этот фонт используется для выделения `имен файлов`, `команд` и `информации, вводимой пользователем`.

1 Введение в LINUX

Linux, возможно, является наиболее значительным достижением в области свободно распространяемых программ со времен Space War, или более позднего Emacs. Он превратился в операционную систему для бизнеса, образования и индивидуального программирования. Linux перестал быть системой для фанатиков-программистов, которые часами сидят перед мерцающими экранами (хотя таких и немало). Эта книга поможет вам извлечь из Linux максимальную пользу.

Linux (произносится "лИнукс") принадлежит семейству UNIX-подобных операционных систем, которая может работать на компьютерах Intel 80386 и 80486. Он поддерживает широкий спектр программных пакетов от TeX до X Windows, компиляторов GNU C/C++, протоколов TCP/IP. Это гибкая реализация ОС UNIX, свободно распространяемая под генеральной лицензией GNU (см. приложение E).

Linux может любой 386 или 486 персональный компьютер превратить в рабочую станцию. Он преподнесет всю мощь UNIX к кончикам ваших пальцев. Бизнесмены инсталлируют Linux в сетях машин, используют операционную систему для обработки данных в сфере финансов, медицины, распределенной обработки, в телекоммуникациях и т.д.

Университеты по всему миру применяют Linux в учебных курсах по программированию и проектированию операционных систем. Разумеется, повсеместно программисты-энтузиасты используют Linux дома для программирования, решения своих прикладных задач и всевозможного хакерства.

Что делает Linux столь отличным от других ОС - это его создание версии UNIX "*на общественных началах*" (*free implementation*).. Он был создан и продолжает совершенствоваться и развиваться группой добровольцев, первоначально в кругу пользователей сети Internet, которые обменивались кодами, информацией об обнаруженных ошибках, выявлением проблем, возникавших при расширении сферы применения. Все желающие приглашаются подключиться к этой работе. Единственное, что требуется - это интерес к семейству UNIX и желание совершенствовать свои навыки в этой сфере. Данная книга - ваш путеводитель.

1.1 Об этой книге

Эта книга является руководством по инсталляции и пособием по начальному знакомству с системой Linux. Цель - приобщить новых пользователей к этой системе и собрать возможно больше существенного материала по ее использованию в одной книге. Вместо того, чтобы перегружать книгу техническими деталями, которые быстро устаревают, мы даем основы, которые помогут вам самостоятельно находить и осваивать дополнительную информацию.

Linux прост в инсталляции и использовании. Но, как и во всякой реализации UNIX, в нем присутствуют элементы "черной магии", которые необходимы при обеспечении его корректной работы. Мы надеемся, что эта книга будет вам хорошим путеводителем по Linux и покажет, насколько простой может выглядеть эта операционная система.

В этой книге мы рассматриваем следующие вопросы:

- Что такое Linux? Особенности структуры и философии этой уникальной операционной системы, и что она может вам дать.
- Все детали, необходимые для практического использования Linux, включая рекомендации по желательной конфигурации аппаратуры.
- Как получить и инсталлировать Linux. Существует много способов распространия программного обеспечения под Linux. Мы описываем общую ситуацию, связанную с его распространением, рассказываем, как его приобрести и инсталлировать. Это издание содержит также специфические инструкции по дистрибуции Linux как Slackware.
- Краткое учебное пособие по UNIX для тех пользователей, которые до этого не встречались с ОС UNIX. Надеемся, что это пособие дает достаточно материала для новичков, чтобы получить базовые знания и начать ориентироваться в этой ОС.
- Введение в системное администрирование Linux. Это покрывает наиболее важные задачи, с которыми следует познакомиться новым администраторам Linux, с

таками задачами как регистрация новых пользователей, управление файловой системой и тому подобное.

- Информация о конфигурировании более продвинутых аспектов Linux, таких как X Window System, сетевая работа с TCP/IP и SLIP, и установке электронной почты.

Эта книга для пользователей персональных компьютеров, желающих начать работать с Linux. Мы не предполагаем предварительного опыта работы с UNIX, но надеемся, что новички будут обращаться по ходу дела к дополнительной литературе. Для незнакомых с UNIX в Приложении A приведен список полезных источников. В общем случае предполагается чтение этой книги совместно с какой-либо книгой по общим концепциям ОС UNIX.

1.2 Краткая история Linux

UNIX - одна из самых популярных в мире операционных систем благодаря тому, что ее сопровождает и распространяет большое число компаний. Первоначально она была создана как многозадачная система для миникомпьютеров и мэйнфреймов в середине 70-ых годов, но с тех пор она выросла в одну из наиболее распространенных операционных систем, несмотря на свой временами обескураживающий интерфейс и отсутствие централизованной стандартизации.

В чем реальная причина популярности UNIX? Многие хакеры нутром чувствуют, что UNIX - это "настоящая вещь", Единственная Настоящая Операционная Система. Отсюда и появление Linux, как системы, разрабатываемой все более расширяющейся группой энтузиастов UNIX, которые хотят собственноручно в ней поковыряться.

Существуют версии UNIX для многих систем, начиная от персонального компьютера, до суперкомпьютеров, таких как Cray Y-MP. Большинство версий UNIX для персональных компьютеров достаточно дороги и сложны. К моменту написания этой книги одномашинная версия AT&T's System V для 386 стоила US$1500.

Linux - свободно распространяемая версия UNIX, первоначально была разработана Линусом Торвальдсом (Linus Torvalds)[1] в Университете Хельсинки (Финляндия). Linux был создан с помощью многих UNIX-программистов и энтузиастов из Internet, тех, кто имеет достаточно навыков и способностей развивать систему. Ядро Linux не использует коды AT&T или какого-либо другого частного источника, и большинство программ Linux разработаны в рамках проекта GNU из Free Software Foundation в Cambridge, Massachusetts. Но в него внесли лепту также программисты всего мира.

[1] (torvalds@kruuna.helsinki.fi)

Первоначально Linux создавался Линусом Торвальдсом как хобби. Его вдохновила операционная система Minix - маленькая UNIX-система, созданная Andy Tanenbaum, и впервые Linux обсуждался по компьютерной сети в рамках USENET newsgroup `comp.os.minix`. В этих обсуждениях прежде всего принимали участие пользователи Minix из учебных и научных заведений, которым хотелось чего-то большего, чем Minix.

Раннее развитие Linux прежде всего было связано с проблемой переключения задач в защищенном режиме для 80386. Все писалось на ассемблере. Линус вспоминает:

> "После этого началось спокойное плавание: по-прежнему беспросветное кодирование, но у меня были различные подсобные программы и отладка была облегчена. На этом этапе я стал использовать Си и это существенно ускорило дело. В это же время я стал серьезно обдумывать маниакальную идею, как сделать Minix лучше себя самого. Я надеялся в один прекрасный день перекомпилировать gcc под Linux..." "Два месяца ушло на написание самых базовых программ, а затем чуть больше времени на драйвер винчестера (с большим количеством ошибок, но все-таки работавшим на моей машине) и простую файловую систему. В результате я подготовил версию 0.01 (примерно конец августа 1991 г.). Она была не слишком изящной, в ней не было драйвера гибких дисков и она многое не могла делать. Но я уже не смог остановиться, пока не создал свой Minix."

Относительно появления Linux версии 0.01 никогда не делалось никаких официальных заявлений. Исходные тексты 0.01 не давали даже нормального выполняемого кода: они фактически состояли лишь из набора заготовок для ядра и молчаливо предполагали, что вы имеете доступ к Minix-машине, чтобы иметь возможность компилировать их и совершенствовать.

5-го октября 1991 года Линус об'явил первую "официальную" версию Linux, версия 0.02. В это время Linux уже мог выполнять `bash` (the GNU Bourne Again Shell) и `gcc` (the GNU C compiler), но мало еще что работало. Вновь это рассматривалось как создание некой хакерской системы. Основное внимание - создание ядра. Никакие вопросы поддержки работы с пользователем, документирования, тиражирования и т. п. даже не обсуждались. Кажется, что и сегодня сообщество Linux-истов считает эти вопросы вторичными по сравнению с "настоящим программированием" - развитием ядра.

Линус писал в `comp.os.minix`:

> "Грустите ли вы по тем прекрасным временам Minix-1.1, когда мужчины были настоящими мужчинами и писали свои собственные драйверы на все

> устройства? У вас сейчас нет под рукой настоящего проекта и вы вымираете от невозможности вонзить свои зубы в какую-то ОС, которую бы можно было модифицировать под свои желания? Не находите ли вы деморализующей ситуацию, когда все в Minix работает? Нет больше бессонных ночей, которые позволяли заставить хитрые программы работать правильно? Тогда это место для вас.” ”Как я уже говорил месяц назад, сейчас я работаю над некоммерческой Minix-подобной ОС для 386-го компьютера. Она уже доведена до такого состояния, когда ею даже можно пользоваться (хотя может быть там не то, что бы вы хотели), и я хочу выложить исходные тексты для широкого распространения. Это версия 0.02, но в ней уже успешно работают `bash`, `gcc`, `gnu-make`, `gnu-sed`, `compress` и т.д.”

После версии 0.03 Линус скачком перешел в нумерации к версии 0.10, так как над проектом стало работать много народу. После нескольких последовавших пересмотров версий, Линус присвоил очередной версии номер 0.95, чтобы тем самым отразить свое впечатление о том, что скоро возможна уже ”официальная” версия. (Обычно программам не дают номер версии 1.0 до того, как она теоретически завершена и отлажена). Это было в марте 1992 г. Примерно через полтора года - в декабре 1993 версия ядра все еще была Linux 0.99.pl14 - асимптотически приближаясь к 1.0. Во время написания книги текущая версия ядра 1.1 patchlevel 52, и на подходе версия 1.2.

Сегодня Linux - это полноценная ОС семейства UNIX, способная работать с X Windows, TCP/IP, Emacs, UUCP, mail и USENET. Практически все важнейшие программные пакеты были поставлены и на Linux, т.е. для Linux теперь доступны и коммерческие пакеты. Все большее разнообразие оборудования поддерживается по сравнению с первоначальным ядром. Многие тестировали Linux на 486-ом и установили, что он вполне сравним с рабочими станциями Sun Microsystems и Digital Equipment Corporation. Кто мог предположить, что этот ”маленький UNIX” вырастет настолько, что сможет делать все в мире компьютеров.

1.3 Системные характеристики

Linux поддерживает большинство свойств, присущих другим реализациям UNIX, плюс ряд тех, которых больше нигде нет. Этот раздел - поверхностный обзор характеристик ядра Linux .

Linux - это полная многозадачная многопользовательская операционная система (точно также как и другие версии UNIX). Это означает, что одновременно много пользователей могут работать на одной машине, одновременно выполнять много программ.

Linux достаточно хорошо совместим с рядом стандартов для UNIX (насколько можно говорить о стандартизации UNIX) на уровне исходных текстов, включая IEEE POSIX.1, System V и BSD. Он создавался имея в виду такую совместимость. Поэтому, скорее всего, вы найдете в Linux черты, присущие многим UNIX-системам. Большинство свободно распространяемых по сети Internet программ для UNIX может быть откомпилировано для LINUX практически без особых изменений. Кроме того, все исходные тексты для Linux, включая ядро, драйверы устройств, библиотеки, пользовательские программы и инструментальные средства распространяются свободно.

Другие специфические внутренние черты Linux включают контроль работ по стандарту POSIX (используемый оболочками, такими как csh и bash), псевдотерминалы (pty), поддержку национальных и стандартных клавиатур динамически загружаемыми драйверами клавиатур.

Linux также поддерживает **виртуальные консоли (virtual consoles)**, которые позволяют "переключать экраны" на консоли в текстовом режиме. Те, кто пользовался программой "`screen`", найдут подобное в реализации виртуальной клавиатуры Linux.

Ядро может само эмулировать команды 387-FPU, так что системы без сопроцессора могут выполнять программы, на него расчитывающие (т.е. с плавающей точкой).

Linux поддерживает различные типы файловых систем для хранения данных. Некоторые файловые системы, такие как файловая система *ext2fs*, были созданы специально для Linux. Поддерживаются также другие типы файловых систем, такие как Minix-1 и Xenix. Реализована также файловая система MS-DOS, позволяющая прямо обращаться к файлам MS-DOS на жестком диске. Поддерживается также файловая система ISO 9660 CD-ROM для работы с дисками CD-ROM. Подробнее о файловых системах говорится в Главах 2 и 4.

Linux обеспечивает полный набор протоколов TCP/IP для сетевой работы. Это включает драйверы устройств для многих популярных карт Ethernet, SLIP (Serial Line Internet Protocol, обеспечивающие вам доступ по TCP/IP при последовательном соединении), PLIP (Parallel Line Internet Protocol), PPP (Point-to-Point Protocol), NFS (Network File System), и так далее. Поддерживается весь спектр клиентов и услуг TCP/IP, таких как FTP, `telnet`, NNTP и SMTP. О сетевых проблемах мы будем говорить в Главе 5.

Ядро Linux сразу создано с учетом специального защищенного режима для процессоров Intel 80386 и 80486. В частности, Linux использует парадигму описания памяти в защищенном режиме и другие новые свойства процессоров. Любой знакомый с защищенным режимом процессора 80386 знает, что этот чип проектировался для многозадачных систем вроде UNIX (или Mulics). Linux использует эти свойства.

Ядро Linux поддерживает загрузку только нужных страниц. То есть с диска в память загружаются те сегменты программы, которые действительно используются. Возможно использование одной страницы, физически один раз загруженной в память, несколькими выполняемыми программами.

Для увеличения об'ема доступной памяти Linux осуществляет также разбиение диска на страницы: то есть на диске может быть выделено до 256 Мбайт "**пространства для свопинга**" (**swap space**).[2] Когда системе нужно больше физической памяти, то она с помощью свопинга выводит неактивные страницы на диск. Это позволяет выполнять более об'емные программы и обслуживать одновременно больше пользователей. Однако свопинг не исключает наращивания физической памяти, поскольку он снижает быстродействие, увеличивает время доступа.

Ядро также поддерживает универсальный пул памяти для пользовательских программ и дискового кэша. При этом для кэша может использоваться вся память, и наоборот, кэш уменьшается при работе больших программ.

Выполняемые программы используют динамически связываемые библиотеки, т.е. выполняемые программы могут совместно использовать библиотечную программу, представленную одним физическим файлом на диске (иначе, чем это реализовано в механизме разделяемых библиотек SunOS). Это позволяет выполняемым файлам занимать меньше места на диске, особенно тем, которые многократно используют библиотечные функции. Есть также статические связываемые библиотеки для тех, кто желает пользоваться отладкой на уровне об'ектных кодов или иметь "полные" выполняемые программы, которые не нуждаются в разделяемых библиотеках. В Linux разделяемые библиотеки динамически связываются во время выполнения, позволяя программисту заменять библиотечные модули своими собственными.

Для обеспечения отладки ядро Linux выдает дампы памяти для "посмертного" анализа. Использование дампа и динамических отладчиков позволяет определить причины краха программы.

1.4 Программные характеристики

В этом разделе мы представим вам многие приложения, доступные в Linux, и поговорим об общих задачах вычисления. В конечном счете - наиболее важным в системе является то, насколько широк спектр доступных в ней программ. А тот факт, что большая часть

---

[2] (Swap space не совсем подходящее имя, в Linux в область своппинга выгружается не весь процесс, а только отдельные его части, в которых нет необходимости).

этих программ распространяется свободно - усиливает впечатление.

1.4.1 Базовые команды и утилиты

Практически любая утилита, которую вы ожидаете найти в стандартных реализациях UNIX, имеется и в Linux. Сюда включены и базовые команды, такие как `ls`, `awk`, `tr`, `sed`, `bc`, `more` и т.д. Назовите любую - она есть в Linux. Поэтому вы в праве ожидать знакомой рабочей UNIX-среды. В Linux есть все стандартные команды и утилиты. (Новички могут посмотреть Главу 3 для начального знакомства с базовыми командами UNIX).

В Linux имеются многие текстовые редакторы, включая `vi`, `ex`, `pico`, `jove`, также как GNU Emacs и его вариации, вроде Lucid Emacs (который содержит расширение для использования под X Windows) и `joe`. Скорее всего, любой текстовый редактор, к которому вы привыкли, перенесен в Linux.

Выбор редактора - явление любопытное. Многие пользователи UNIX до сих пор используют "простые" редакторы вроде `vi` (кстати, автор писал эту книгу в Linux, используя редактор `vi`) (кстати, переводчик переводил эту книгу в Linux, используя редактор `red`).

Но `vi` имеет много ограничений по причине своего преклонного возраста, сейчас завоевывают популярность более современные и сложные редакторы вроде Emacs. Emacs поддерживает базирующийся на LISP макроязык и интерпретатор, мощный командный синтаксис и другие расширения. Существуют макропакеты Emacs, позволяющие читать электронную почту и новости, редактировать содержимое каталогов и даже проводить сеансы психотерапии с использованием искусственного интеллекта (неоценимая возможность для измотанных Linux-ом хакеров).

Интересное замечание - большинство утилит Linux имеют статус GNU. Эти утилиты часто поддерживают наиболее современные черты, не содержащиеся в стандартных версиях BSD или AT&T. Например, версия GNU редактора `vi` - `elvis`, содержит структурный макроязык, который отличается от исходной реализации AT&T. Но тем не менее, утилиты GNU сохраняют совместимость с их тезками из BSD и System V. Многие считают, что GNU версии лучше исходных программ.

Многие пользователи самой важной утилитой считают **shell**. shell - это программа, которая читает и выполняет команды пользователя. Кроме того, многие оболочки (shells) имеют такие возможности, как **контроль выполнения** (**job control**) (позволяя пользователю управлять несколькими параллельными процессами), перенаправление входа-выхода и командный язык для написания **командных файлов** (**shell scripts**).

Командный файл - это программа на языке оболочки, аналогичная "batch file" в MS-DOS.

В Linux много типов оболочек. Наиболее важное различие между ними - используемый командный язык. Например, **C Shell** (`csh`) использует командный язык, чем-то напоминающий язык программирования Си. Классический **Баурновский shell** (**Bourne Shell**) использует иной командный язык. Обычно выбор оболочки обусловлен выбором соответствующего командного языка. Выбранная оболочка в какой-то мере определяет вашу рабочую среду.

Не важно, к какой оболочке вы привыкли, та или иная ее версия есть в Linux. Наиболее популярная оболочка - это GNU Bourne Again Shell (`bash`), т.е. вариант Bourne shell, включающий много современных свойств и возможностей, таких как управление работами, командную историю, дописывание имен команд и имен файлов, Emacs-подобный интерфейс редактирования командной строки и мощное расширение стандартной оболочки (Bourne shell).

Другая популярная оболочка - `tcsh`, версия C Shell с более современными функциями по сравнению с `bash`. Другие оболочки: `zsh` - небольшая баурно-подобная оболочка; `ksh` - оболочка Корна; `ash` - оболочка BSD и `rc` -оболочка проекта Plan 9.

Что особенно важно сказать относительно этих оболочек? Linux дает вам уникальную возможность кроить систему под ваши личные нужды. Например, если вы единственный пользуетесь этой системой и вы предпочитаете редактор vi и bash в качестве оболочки, то нет необходимости иметь прочие редакторы и оболочки. "Сделай сам, как тебе нравится" - это позиция хакеров и пользователей Linux.

1.4.2 Обработка текстов и слов

Почти все пользователи нуждаются в какой-либо системе подготовки документов. (Много ли вы знаете энтузиастов компьютерной обработки, которые все еще пользуются ручкой и бумагой? Мы догадываемся, что очень немного). В мире персональных компьютеров *обработка слов* (*word processing*) - норма: она включает редактирование и манипуляции с текстом (часто в стиле WYSIWYG - "What-You-See-Is-What-You-Get" - "Что-Вы-Видите-ЕстьТо- Что-Вы-Имеете" - ЧВВЕТЧВИ) и получение печатных копий, содержащих рисунки, таблицы и другой гарнир.

В мире UNIX *обработка текстов* (*text processing*)- вещь более естественная, чем классическая концепция обработки слов. Вместо того, чтобы вызывать специальные средства обработки слов, исходный текст можно модифицировать любым текстовым редактором, таким как vi или Emacs. После подготовки текста пользователь форматирует текст специальными программами, преобразующими его к нужному для печати виду. Это в чем-то аналогично программированию на языке вроде Си, и "компилированию" текста в пригодную для печати форму.

В Linux много текстовых процессоров. Один из них `groff` - GNU версия классического форматера текстов `nroff`, первоначально созданного в Bell Labs и до сих пор используемого во многих UNIX. Другой современный текстовый процессор - TeX, создан знаменитым в компьютерном мире Дональдом Кнутом (Donald Knuth). Доступны диалекты TeX, такие как LaTeX.

Текстовые процессоры, такие как TeX и `groff` значительно различаются по синтаксису языков форматирования. Предпочтение той или иной системы форматирования в значительной мере базируется на том, какие имеются вспомогательные утилиты и насколько система вам по вкусу.

Например, некоторые люди считают `groff` несколько заумным, поэтому они используют TeX, который более понятен для хомо сапиенс. Между тем, `groff` может давать ясный ASCII выход, легко читаемый на терминале, в то время как TeX прежде всего предназначен для вывода на печать. Но существуют многочисленные программы, позволяющие получить читаемый текст из отформатированных с помощью TeX документов или конвертирующих TeX, например, в `groff`.

Другой текстовый процессор `texinfo` - расширение TeX, используемое для подготовки программной документации в Free Software Foundation. `texinfo` может формировать печатный документ или гипертекст "Info" для просмотра на экране на основе одного исходного файла. Файлы Info - это основной формат для документирования, используемый в GNU, в частности в Emacs.

Текстовые процессоры широко используются в компьютерном мире для подготовки статей, диссертаций, статей для журналов и книг (типографский вариант этой книги был подготовлен с использованием LaTeX). Возможность обрабатывать исходный язык как текстовый файл открывает двери.

Как выглядит язык форматирования? В общем случае он содержит сам текст с "управляющими кодами" для производства заказанных действий, таких как изменение фонтов, установление полей, формирование страниц и т.д.

В качестве примера возьмем следующий текст:

```
Mr. Torvalds:

We are very upset with your current plans to implement
post-hypnotic suggestion in the Linux terminal driver code.
We feel this way for three reasons:
```

```
1.  Planting subliminal messages in the terminal driver is not
  only immoral, it is a waste of time;

2.  It has been proven that ``post-hypnotic  suggestions´´ are
  ineffective when used upon unsuspecting UNIX hackers;

3.  We have already implemented high-voltage electric  shocks,
  as a security measure, in the code for login.


We hope you will reconsider.
```

Этот текст на языке форматирования LaTeX будет выглядеть следующим образом:

```
beginquote
Mr. Torvalds:

We are very upset with your current plans to  implement em post-
hypnotic
suggestion/      in      the       bf      Linux       termi-
nal driver code. We feel this
way for three reasons:
beginenumerate
item    Planting    subliminal    messages    in    the    ker-
nel driver is not only
    immoral, it is a waste of time;
item  It   has   been   proven   that   ``post-hypnotic   sugges-
tions´´ are ineffective
    when used upon unsuspecting UNIX hackers;
item  We   have   already  implemented  high-voltage   elec-
tric shocks, as a
    security measure, in the code for tt login.
endenumerate
We hope you will reconsider.
endquote
```

Автор входит в ”исходный” текст, приведенный выше, используя любой текстовый редактор, и генерирует форматированный выход, обрабатывая текст с помощью LaTeX. На первый взгляд такой язык может показаться достаточно заумным, но в действительности он очень прост в освоении. Использование текстового процессора обеспечивает поддержку типографских стандартов. Например, все перечисленные в рамках документа страницы будут выглядеть одинаково. Пишущий может сосредоточиться на тексте, а не на том, как это следует оформлять с точки зрения типографских требований.

Процессоры слов типа WYSIWYG привлекательны по многим причинам: они обеспечивают мощный (а иногда и сложный) визуальный интерфейс для редактирования документов. Но этот интерфейс традиционно ограничен желательными и приемлемыми для пользователя формами выдачи. Так, многие процессоры имеют средства подготовки к печати математических формул.

Преимущество текстовых процессоров состоит в том, что они позволяют описывать именно то, что вы хотите. Они позволяют также редактировать исходный текст любым текстовым редактором и затем конвертировать в различные форматы. Платой за такую гибкость является отсутствие тех качеств интерфейса, которые есть у WYSIWYG.

Многие пользователи процессоров слов привыкли видеть отредактированный текст сразу в процессе редактирования. С другой стороны, при использовании текстового процессора пишущий не заботится о том, как будет выглядеть уже отформатированный текст. Пишущий лишь должен иметь представление о том, какие действия произведут вставляемые им команды форматирования.

Есть программы, которые позволяют посмотреть на графическом дисплее вид отформатированного документа перед его печатью. Например, программа `xdvi` отображает "независимый от устройства" файл, сгенерированный TeX под X Window. Другое приложение - `xfig`, обеспечивает графический интерфейс WYSIWYG для рисунков и диаграмм, который в последующем конвертируется в команды текстового процессора, включаемые в документ.

Следует напомнить, что текстовые процессоры, вроде `nroff`, появились задолго до процессоров слов. Многие предпочитают использовать текстовые процессоры из-за их гибкости и независимости от графической среды. В любом случае, в Linux имеется процессор слов idoc, а также скоро ожидается появление коммерческих процессоров слов. Но если вы не хотите расставаться с процессором слов, к которому вы привыкли в MS-DOS, вы всегда можете использовать MS-DOS или другую операционную систему в дополнение к Linux.

Существует и много других утилит текстовых процессоров. Мощная система METAFONT используется для проектирования фонтов в TeX. Другие программы включают `ispell` - интерактивный контролер правописания, `makeindex` - используется для генерации индексов в документах, подготовленных в LaTeX. Существует много макропакетов для `groff` и TeX для форматирования документов различных типов и математических текстов, а также тьма конверторов, транслирующих TeX или `groff` во многие другие форматы.

1.4.3 Языки программирования и утилиты

Linux обеспечивает полную UNIX-среду программирования, включая все стандартные библиотеки, программный инструментарий, компиляторы, отладчики, которые вы встречаете и в других UNIX-системах. В мире UNIX большинство приложений и системных программ делаются на Си или Си++. Стандартным компилятором для Си и Си++ в Linux служит GNU `gcc`, который является современным компилятором, поддерживающим много опций. Он способен компилировать Си++ (включая особенности AT&T 3.0 features) также, как Objective-C, другие об'ектно-ориентированные диалекты Си.

Кроме Си и Си++ многие другие компиляторы и интерпретаторы были перенесены в Linux, такие как Smalltalk, FORTRAN, Pascal, LISP, Scheme и Ada (если вы настолько мазохист, чтобы программировать на Аде - мы не будем вам препятствовать). В дополнение, существуют различные ассемблеры для написания кодов для защищенного режима 80386, а также любимые хакерами языки, вроде Perl (язык сценариев, как окончательный тупик или вершина для всех языков такого типа) и Tcl/Tk (shell-подобный командный язык, включающий поддержку разработки простейших приложений в X Window).

В Linux был перенесен продвинутый отладчик `gdb`, позволяюший пошагово выполнять программы в поисках ошибок или анализировать крах программ с помощью дампов памяти. `gprof` - утилита профилирования, показывающая, где ваша программа при выполнении тратит больше времени. Текстовый редактор Emacs позволяет осуществлять интерактивное редактирование. Другие инструменты, включая GNU `make` и `imake` используются для управления компиляцией больших программ; RCS - система для защиты и сопровождения исходных текстов.

Linux содержит динамические библиотеки (DLL), которые позволяют экономить место, поскольку они вызываются только во время выполнения. Эти библиотеки позволяют также прикладному программисту переопределять функции, включая свои коды. Например, если программист желает написать свою собственную версию библиотечной программы `malloc()`, компоновщик подключит новую программу вместо библиотечной.

Linux идеален для создания UNIX-приложений. Он обеспечивает современное программное окружение со всеми дополнительными погремушками. Поддерживаются различные стандарты вроде POSIX.1, позволяющие легко переносить программы, написанные для Linux, на другие системы. Профессиональные UNIX-программисты и системные администраторы могут использовать Linux для домашних компьютеров, а с них переносить написанные программы на компьютеры своей фирмы. Это может не только сэкономить много времени и денег, но и обеспечить комфортабельную работу на домашнем компьютере.[3] Студенты, изучающие компьютерные науки, могут использовать Linux для обучения программированию в UNIX и изучения таких аспектов, как архитектура ядра.

Через Linux вы не только имеете доступ к полному набору библиотек и утилит, но также к исходным текстам ядра и библиотек.

1.4.4 Система X Window

Система X Window (или кратко просто X) - стандартный графический интерфейс для UNIX-машин. Это мощная среда, поддерживающая много приложений. Используя X Window, пользователь может одновременно иметь на экране несколько окон, при этом каждое имеет независимый login. Часто используется мышь, хотя она необязательна.

Было написано много специфических X-приложений, таких как игры, графические утилиты, инструментарий для программирования и документирования и т.д. С Linux и X ваш компьютер - замечательная рабочая станция. Используя протоколы TCP/IP, вы можете смотреть у себя X-приложения, выполняемые на других машинах.

Система X Window была первоначально создана в MIT и свободно распространялась. Существует много и коммерческих приложений, расширяющих возможности X Window. Для Linux есть система X Window, известная как XFree86; версия X11R5 свободно распространяется для UNIX-систем типа Linux. XFree86 поддерживает широкий спектр видео устройств, включая VGA, Super VGA, различные видео адаптеры с ускорителями. Это полный комплект X Window, содержащий сам сервер, много прикладных программ и утилит, программные библиотеки и документацию.

Стандартные X-приложения включают `xterm` (эмулятор терминала, используемый в большинстве текстовых приложений в X Window); `xdm` (X-менеджер, обслуживающий login); `xclock` (представление простых часов); `xman` (X-ориентированное руководство по Linux) и т.д. Трудно перечислить все приложения X, доступные в Linux, но базовый комплект XFree86 включает "стандартные" приложения, содержащиеся в исходной версии MIT. Но доступно и многое другое, теоретически, все написанное для X Window должно прямо компилироваться и для Linux.

Интерфейс X Window в большой степени контролируется **менеджером окон** (**window manager**). Эта программа отвечает за размещение окон, изменение их размеров, размещение иконок, перемещение окон, вид оконных рамок и т.д. Стандартный дистрибутив XFree86 включает twm, классический оконный менеджер MIT, но также

---

[3] (Автор использует дома Linux для разработки и тестирования X Window приложений, которые могут прямо транслироваться на любых рабочих станциях).

имеются и более современные менеджеры, такие как Open Look Virtual Window Manager (`olvwm`). Среди пользователей Linux популярен `fvwm`. Это небольшой менеджер окон, требующий в два с лишним раза меньше памяти, чем `twm`. Он обеспечивает трехмерное представление обрамления окон и виртуальный рабочий стол (desktop) - если пользователь подвигает мышь к краю экрана, все изображение смещается, будто дисплей имеет большие размеры, чем на самом деле. fvwm более традиционен и позволяет реализовать все функции доступа как с клавиатуры, так и от мыши. Многие дистрибутивы Linux содержат `fvwm`, как стандартный менеджер окон.

Дистрибутив XFree86 содержит программные библиотеки и включает файлы для тех программистов, кто желает создавать приложения в X. Поддерживаются различные множества widget (графических представлений), такие как Athena, Open Look и Xaw3D. Включены все стандартные фонты, битмэпы и документация. Поддерживается также PEX (программный интерфейс для трехмерной графики).

Многие пользующиеся X-ом используют и имеющиеся в Motif наборы widget. Несколько компаний продают одно- и многопользовательские лицензии бинарников Motif в Linux. Поскольку Motif сам по себе сравнительно дорог, немногие владельцы Linux имеют Motif. Тем не менее, бинарники, статически связанные с библиотечными программами Motif, могут свободно распространяться. Если вы написали программы с использованием Motif и хотите их передавать, вы должны позаботиться о самодостаточности кодов.

Главные ограничения использования X Window происходят от требований к аппаратуре. Минимально необходим 386 процессор с 4 Мбайт RAM. Но для более комфортного режима желательно не менее 8 Мбайт. Желательно и процессор побыстрее, но прежде всего необходима память. Для действительно хорошего результата лучше иметь карту с акселератором (как например S3-chipset). На Linux с XFree86 был достигнут рейтинг выполнения, превосходящий 140000 xstones. На приличном компьютере вы можете убедиться, что X под Linux работает не хуже, или даже быстрее, чем на других UNIX.

В Главе 5 мы обсудим вопросы инсталляции и использования X.

1.4.5 Работа в сети

Интересует ли вас связь с миром? Да? Нет? Может быть? Linux поддерживает два базовых сетевых протокола UNIX: **TCP/IP** и **UUCP**. TCP/IP (Transmission Control Protocol/Internet Protocol) есть множество сетевых парадигм, позволяющих системам по всему миру связываться по единой сети, известной как Internet. С помощью Linux, TCP/IP и подключения к сети вы можете общаться с пользователями и машинами всего Internet через электронную почту, новости USENET, передачу файлов FTP и т.п.

В Internet много машин под Linux.

Большинство сетей TCP/IP используют Ethernet, как физическое транспортное средство. Linux поддерживает многие популярные карты Ethernet и интерфейсы.

Однако, поскольку не у всех есть дома плата Ethernet, Linux также поддерживает **SLIP** (Serial Line Internet Protocol), позволяющий связываться с Internet через модем. Для использования SLIP вы должны иметь доступ к SLIP-серверу, машине связанной с сетью и обеспечивающей вам вход в Internet. Многие фирмы и университеты предоставляют SLIP-сервис. Если ваш Linux имеет Ethernet и модем, вы можете сконфигурировать систему как SLIP-сервер для других хостов.

NFS (Network File System) позволяет вам использовать файлы совместно с другими машинами сети. FTP (File Transfer Protocol) позволяет передавать файлы между машинами. Другие приложения включают `sendmail` - систему передачи и получения электронной почты с использованием протокола SMTP; базирующуюся на протоколе NNTP, системе электронных новостей типа C-News и INN; `telnet`, `rlogin` и `rsh` - позволяют войти и выполнить команды на других машинах сети; `finger` - позволяет получать информацию о других пользователях Internet. Фигурально выражаясь - существуют тонны различных приложений для протокола TCP/IP.

Полный спектр различных программ для чтения почты и новостей существует в Linux, это, например, `elm`, `pine`, `rn`, `nn` и `tin`.

Если у вас есть опыт работы с приложениями TCP/IP на других UNIX-системах, Linux не будет для вас чем-то новым. Система обеспечивает стандартный программный интерфейс, поэтому любая программа, использующая TCP/IP, может быть легко перенесена на Linux. X-сервер Linux также поддерживает TCP/IP, позволяя отображать выполняемые на других машинах прикладные программы на вашем дисплее.

В Главе 5 мы обсудим конфигурацию и установку для Linux TCP/IP, включая SLIP.

UUCP (UNIX-to-UNIX Copy) - старейший механизм передачи файлов, электронной почты и электронных новостей между UNIX-машинами. Классически, UUCP-машины связываются друг с другом по телефонным линиям через модем, но UUCP может использовать в качестве транспортного средства и связь по TCP/IP. Если у вас нет доступа по TCP/IP или SLIP-сервера, вы можете сконфигурировать свою систему так, чтобы посылать и получать файлы и электронную почту с использованием UUCP. Подробнее смотрите в Главе 5.

1.4.6 Телекоммуникации и BBS

Если у вас есть модем, вы можете связываться с другими машинами, используя телекоммуникационные пакеты, имеющиеся в Linux. Многие используют программы телекоммуникации для связи с BBS (Bulletin Board Systems), а также и с коммерческими он-лайновыми системами, вроде Prodigy, CompuServe и America On-Line. Другие через модемы связываются с UNIX-системой в школе или на работе. Вы можете использовать модем и Linux для посылки и приема факсов. Телекоммуникационные пакеты Linux очень похожи на имеющиеся в MS-DOS или других операционных системах.

Один из наиболее популярных телекоммуникационных пакетов в Linux - Seyon - X-приложение, предоставляющее традиционный эргономичный интерфейс со встроенной поддержкой различных протоколов передачи файлов, таких как Kermit, ZModem и т.п. Есть также телекоммуникационные программы C-Kermit, `pcomm` и `minicom`. Это напоминает наборы телекоммуникационных программ в других системах.

Если у вас нет доступа к SLIP-серверу, вы можете использовать `term` для мультиплексирования вашей последовательной линии. `term` обеспечит вам множественный доступ через модем на удаленную машину. term также позволит перенаправлять X-клиента на локальный X-сервер через последовательную линию, давая возможность отобразить удаленное X-приложение на вашей Linux-системе. Другой пакет - KA9Q - обеспечивает интерфейс, похожий на SLIP.

BBS - это сегодня хобби многих программистов. Linux поддерживает большое разнообразие программ для BBS, большинство из которых более мощные, чем в других операционных системах. С телефонной линией, модемом и Linux вы можете превратить ваш компьютер в BBS, обеспечив dial-in доступ к своей системе для пользователей с Земного шара. Программное обеспечение BBS для Linux включает XBBS и пакеты UniBoard BBS.

Большинство программ BBS ограничивают пользователя меню-системой, где имеется некоторый фиксированный набор функций. Альтернативой доступу в BBS служит полный спектр возможностей доступа UNIX, который позволяет вам работать с удаленной машиной на правах обычного пользователя.

Если у вас нет возможностей использовать TCP/IP или UUCP, Linux позволяет также связываться с рядом BBS, таких как FidoNet по телефонным линиям и обмениваться новостями и почтой. Дополнительную информацию можно найти в Главе 5.

1.4.7 Интерфейс с MS-DOS

Существуют различные утилиты для связи с миром MS-DOS. Наиболее известен Linux MS-DOS Emulator, позволяющий выполнять многие MS-DOS программы прямо на Linux. Несмотря на то, что Linux и MS-DOS абсолютно различные операционные системы, среда защищенного режима для 80386 позволяет некоторым задачам вести себя так, как это делают прикладные программы MS-DOS.

Эмулятор MS-DOS все еще в стадии совершенствования, но многие популярные пакеты в нем уже выполняются. Понятно, что некоторые приложения MS-DOS, использующие специфические или скрытые свойства системы, никогда не будут выполняться, поскольку эмулятор не знает, как их эмулировать. Например, вы не сможете без шероховатостей выполнять программы, использующие свойства защищенного режима 80386, такие как Microsoft Windows (в расширенном режиме 386-го). Приложения, которые успешно работают под Linux MS-DOS Emulator включают: 4DOS (интерпретатор команд), Foxpro 2.0, Harvard Graphics, MathCad, Stacker 3.1, Turbo Assembler, Turbo C/C++, Turbo Pascal, Microsoft Windows 3.0 (в *реальном* режиме) и WordPerfect 5.1. Стандартные команды и утилиты MS-DOS (такие как PKZIP и т.п.) также работают с эмулятором.

Эмулятор MS-DOS прежде всего предназначается для тех, кому MS-DOS нужен только для выполнения нескольких приложений, но в основном используется Linux. Эмулятор, это не полное повторение MS-DOS. Разумеется, если эмулятор не удовлетворяет вашим потребностям, вы можете использовать MS-DOS непосредственно, как и Linux, на одной и той же машине. При использовании загрузчика LILO можно во время загрузки указать, какую загрузить операционную систему. Linux может сосуществовать с другими операционными системами, с той же OS/2.

Linux обеспечивает "гладкий" интерфейс для обмена файлами между Linux и MS-DOS. Вы можете примонтировать раздел MS-DOS или гибкий диск под Linux и иметь прямой доступ к файлам MS-DOS, как и к "родным".

В настоящее время находится в работе проект, известный как **WINE** - эмулятор Microsoft Windows для X Window System под Linux. По завершению WINE, пользователи будут иметь возможность выполнять MS-Windows приложения прямо из Linux. Это похоже на эмулятор Windows от Sun Microsystems. В момент написания этих строк WINE все-еще на ранней стадии создания, но имеет хорошие перспективы. В Главе 5 мы поговорим об инструментарии MS-DOS, доступном из Linux.

1.4.8 Другие приложения

В Linux огромное количество всевозможных приложений, что и следует ожидать от такой "разносторонней" операционной системы. Основная ориентация Linux была на персональные UNIX-вычисления, но она быстро меняется. Все больше его используют в бизнесе и обучении, все больше появляется на рынке всевозможных коммерческих приложений.

В Linux доступно несколько реляционных баз, включая Postgres, Ingres, и Mbase. Это полномасштабные профессиональные системы управления базами данных типа клиент-сервер, похожие на имеющиеся на других платформах UNIX. Имеется также коммерческая база /`rdb`.

Прикладные научные пакеты включают `FELT` (Finite Element Analysis Tool); `gnuplot` (анализ данных и черчение); `Octave` (пакет символических вычислений, похожий на MATLAB); `xspread` (калькулятор типа spreadsheet); `xfractint` (X-вариант популярного рекурсивного генератора Fractint); `xlispstat` (пакет статистики) и многое другое. Другие приложения содержат `Spice` (проектирование и анализ цепей) и `Khoros` (аналого/цифровая обработка сигналов и визуализация).

Разумеется, есть еще много приложений, которые были или будут перенесены на Linux. Linux обеспечивает полный программный UNIX-интерфейс, удобный в качестве исходной базы для любых приложений в любой научной области.

Как и другие операционные системы, Linux не стоит в стороне от компьютерных игр. Это и классические текстовые "подземельные" игры, вроде Nethack и Moria; игры типа MUDs (Multi-User Dungeons, которые позволяют взаимодействовать многим пользователям), вроде DikuMUD и TinyMUD; а также тьма игр в X, таких как `xtetris`, `netrek` и `Xboard` (X11-версия gnuchess). Популярная игра типа перестреляй-их-всех-в-лабиринтах - Doom также перенесена в Linux.

Для меломанов Linux поддерживает различные саунд-карты, вроде CDplayer (программа, которая может управлять драйвером CD-ROM, как традиционным CD-плейером), MIDI последовательности и редакторы (позволяющие создавать музыку на синтезаторе или другом, управляемом MIDI инструменте) и саунд-редакторы цифровой записи.

Вы не можете найти нужную прикладную программу? Карта программ Linux, приведенная в приложении A имеет большой список пакетов, которые были написаны для Linux или перенесены в него. Список далеко не полный, хотя и перечисляет большое количество пакетов. Другой способ поиска приложений в Linux, это просматривать файлы INDEX на FTP серверах под Linux, если вы имеете доступ в Internet. Оглянитесь вокруг и вы найдете много того, с чем интересно повозиться.

Но если вы совершенно не можете найти того, что вам надо, вы всегда можете попытаться перенести нужные приложения в Linux. Большая часть свободно распространяемых программ для UNIX могут быть откомпилированы для Linux, как правило, без больших проблем. Или, если компиляция не проходит, вы сами можете написать соответствующую программу. Если вам нужна коммерческая программа, возможно, что существует ее свободно распространяемый вариант. А может, вы убедите компанию сделать выполняемые версии для Linux. История знает случаи, когда удавалось уговорить компании.

1.5 Относительно Copyright для Linux

Общедоступная Лицензия GNU (**the GNU General Public License**) или кратко **GPL**. *GPL* была разработана для проекта GNU ассоциацией Free Software Foundataion. Она устанавливает некоторые положения относительно распространения и модификации "свободнораспространяемых программ". В данном случае "свобода" относится именно к Свободе, а не к стоимости. *GPL* всегда был источником недопонимания и мы надеемся, что этот обзор поможет вам понять цели и задачи *GPL* и его влияние на Linux. Полная копия *GPL* включена в Приложение E.

Первоначально Линус Торвальдс выпустил Linux под лицензией более ограничивающей, чем *GPL*, которая разрешала свободное распространение и модификацию, но запрещала любые денежные расчеты при передаче и использовании. С другой стороны *GPL* позволяет людям продавать и иметь прибыль со свободно распространяемых пограмм, но но не разрешает ограничивать права других в распространении этих программ любым образом.

Прежде всего следует об'яснить, что "свободнораспространяемые программы", под лицензией *GPL* - это не *public domain*. Программы *public domain* - это программы не защищенные с помощью copyright и, фигурально выражаясь, принадлежат "почтенной публике" - обществу. Программы, защищаемые *GPL*, наоборот, защищают авторские права автора или авторов. Это значит, что программы защищены стандартными международными законами copyright, и что автор программ официально обозначен. Так что из факта свободного распространения программ не следует, что они - *public domain*.

Программы под лицензией *GPL* не являются также *shareware*. В общем случае программы *shareware* принадлежат и копируются автором, а автор требует присылать деньги за использование программы после ее передачи. А программы под *GPL* могут распространяться бесплатно.

GPL также позволяет людям брать и модифицировать программы, а также распространять свои собственные версии программ. Однако всякая производная работа, основанная на программах, защищенных должна быть под защитой *GPL*. Другими словами, компания не может, взяв и модифицировав Linux, продавать его под ограничительной лицензией. Любые программы, производные от, должны быть также защищены *GPL*.

GPL позволяет распространять и использовать программы бесплатно. Однако, она позволяет человеку или компании распространять программы *GPL* за деньги и даже получать прибыль от продажи и распространения. Однако, при продаже программ под *GPL* дистрибутор не может лишить таких прав (свободного распространения) покупателя. То есть, если вы купили где-то программы под *GPL*, вы можете их свободно распространять, или сами заняться торговлей ими.

Сначала это может звучать как противоречие. Как это так, продавать с выгодой для себя программы, когда *GPL* позволяет любому иметь их бесплатно? Например, предположим, что какая-то компания решила собрать большое число свободнораспространяемых программ на CD-ROM и занятся их распространением. Эта компания должна вернуть деньги для покрытия расходов на производство и дистрибуцию CD-ROM, компания может также решить сделать на этих продажах прибыль. Это разрешается в соответсвии с *GPL*.

Организация, продающая свободнораспространяемые программы, должна следовать определенным ограничениям, выдвигаемым *GPL*. Первое, они не имеют права ограничить права пользователей, купивших программу. Это значит, что если вы купили CD-ROM с программами под *GPL*, вы можете бесплатно их копировать и свободно распространять этот CD-ROM, или тоже продавать. Второе, дистрибуторы должны доступным образом доводить до пользователей информацию о том, что эти программы находятся под защитой *GPL*. Третье, дистрибуторы должны поставить бесплатно полный исходный код распространяемых программ. Это позволит любому, кто купит программы под *GPL*, модифицировать их.

Позволить
компаниям распространять и продавать свободнораспространяемые программы - вещь очень хорошая. Не всякий имеет доступ к Internet, чтобы скачать программы, вроде Linux, бесплатно.[4] *GPL* позволяет компаниям продавать и распространять программы среди тех, кто не имеет свободного доступа к программам. Например,

---

[4] (прим. переводчика: Правда, сама IP-связь для многих в нашей стране очень и очень даже не бесплатно - во много раз дороже, чем это обходится "среднему американцу". О качестве связи и говорить не хочется).

многие организации продают Linux на дискетах, лентах или CD-ROM по почтовым заказам и делают на этом свою прибыль. Разработчики Linux могут никогда не увидеть какую-либо прибыль для себя от этих продаж. Вот каким образом регулируются отношения между разработчиками и дистрибуторами, когда программы находятся под лицензией *GPL*. Другими словами, Линус неспутайтеавторасназваниемОС*Linux* знал, что компании могут захотеть продавать Linux, и что он может не увидеть и пенни от этих продаж.

В мире свободнораспространяемых программ важным элементом являются деньги. Цель свободнораспространяемых программ - это всегда создание и распространение фантастических программ; чтобы любой мог их свободно достать и использовать. В следующем разделе мы обсудим, как это соотносится с разработкой Linux.

1.6 Проектирование и философия Linux

Когда новый пользователь сталкивается с Linux, часто возникают ложные ожидания. Linux - уникальная операционная система, и важно понимать его философию и особенности проектирования, чтобы эффективно его использовать. Даже если вы умудренный годами UNIX-гуру, вы найдете в последующем интересное для себя.

В фирмах, разрабатывающих коммерческие UNIX, вся система создается под жестким контролем качества, существует система управления написанием программ, внесением изменений, документированием, информированием о выявленных ошибках и их устранением. Разработчикам запрещено по собственному желанию добавлять какие-то свойства или менять критически важные коды по своему желанию. Они могут вносить изменения только, как реакцию на выявленные ошибки, документировать вносимые изменения так, чтобы можно было систему при необходимости "вернуть назад". Каждый разработчик закреплен за одной или несколькими частями системного кода, и только этот разработчик имеет право исправлять замеченные ошибки.

Внутри фирм департаменты контроля качества осуществляют жесткое тестирование всякой новой версии операционной системы. Разработчики обязаны под контролем устранять выявленные ошибки. Существует сложная система статистического анализа, определяющая, сколько ошибок должно быть устранено, чтобы объявить переход к новой версии.

Подход, используемый создателями коммерческого UNIX при написании и сопровождении кодов, весьма сложен и это вполне обоснованно. Фирма должна иметь постоянные и хорошо детализированные подтверждения тому, что следующая версия операционной системы созревает для выпуска на рынок. Отсюда сбор и анализ статистики о работе этой системы. Это очень большая и трудоемкая работа - создавать коммерческий UNIX. Часто настолько большая, что требуются сотни, если не тысячи программистов, специалистов по тестированию, писателей документации, административного персонала. Разумеется, никакие два производителя коммерческого UNIX не похожи друг на друга, но общие принципы именно таковы, как описаны выше.

Применительно к Linux вы можете выкинуть из головы вышеописанную концепцию организации разработки большой программной системы, отладки, контроля качества, статистического анализа и т.п. Судя по всему, Linux был и останется хакерской системой.[5]

Linux первоначально создавался группой энтузиастов в Internet со всего мира. Любой, в Internet и за его пределами, имеющий достаточные знания и навыки, имеет возможность принять участие в совершенствовании и отладке ядра, переносе в Linux новых программ, написании документации, помощи новичкам. Нет определенной организации, отвечающей за развитие системы. Большей частью Linux-сообщество общается через группы по интересам USENET. Существует ряд соглашений для принимающих участие в разработках: например, любой, желающий, чтобы его код был включен в "официальное" ядро, должен написать Линусу Торвальдсу, который проведет тестирование и включит код в ядро (если предлагаемый код вписывается в систему и не противоречит ее принципам - скорее всего он будет включен).

Сама по себе система проектируется по открытому принципу. Хотя число вносимых радикальных изменений в систему уменьшается, общая тенденция выдачи новой версии ядра через несколько месяцев (а иногда и чаще) сохраняется. Разумеется, это приблизительная характеристика, она зависит от многих факторов, включая число обнаруженных ошибок, число замечаний пользователей, и то, сколько часов Линус спал в этом месяце.

Естественно, не все ошибки выявляются и не все проблемы решаются между выпусками версий. Но когда создается впечатление, что существенные и часто проявляющиеся ошибки устранены и система ведет себя достаточно "стабильно", выпускается новая версия. Это не попытка выпустить безошибочную версию, а желание выпустить версию UNIX, с которой можно работать. Linux прежде всего ориентирован на разработчиков.

Все, у кого есть новые программы, которые они хотели бы добавить в систему, обычно делают их доступными для других на "альфа" стадии, т.е. на стадии тестирования теми отважными или еще не уставшими пользователями, которые хотят сокрушать

---

[5] (Что я понимаю под словом "хакер" - это фанатически преданный программированию человек, которому нравится работать на компьютере и делать на нем интересные вещи. Это не соответствует пониманию некоторыми слова "хакер", как обозначения для компьютерного хулигана).

возникающие проблемы первоначального кода. Поскольку Linux-сообщество в большой степени кучкуется вокруг Internet, альфа-программы выкладываются на один или более Linux FTP-сервера (смотрите Приложение C) и посылается письмо в одну из Linux-групп USENET, о том как можно получить и тестировать представленный код. Пользователи, которые скачивают и тестируют эти альфа-программы, могут по почте сообщать результаты, указывать ошибки, задавать вопросы автору.

После решения начальных проблем с альфа-кодом, код приобретает статус "бета", при котором он обычно уже достаточно стабилен, хотя и небезгрешен (он работает, но не все еще его ветви полностью работоспособны). Иначе код попадает в разряд "готового", когда он считается полным и правильным. Относительно новых кодов ядра необходимо просить Линуса включить их в стандартное ядро или добавить как факультативную опцию ядра.

Следует иметь в виду, что это лишь соглашения, а не правила. Некоторые люди так уверены в своих программах, что не считают нужным публиковать альфа-версию. Так что от разработчика зависит в известной мере и процедура.

Вас может несколько удивить столь неупорядоченная система привлечения добровольцев, программирования и отладки UNIX. Может ли она дать вообще положительный результат? Как оказывается, это один из самых эффективных и успешных проектов, когда-либо существовавших. Полностью все ядро было написано без привлечения каких либо частей ранее существовавшего кода. Большая работа была проделана добровольцами по переносу свободно распространяемых программ в Linux. Были написаны библиотеки, создана файловая система, драйверы для многих популярных устройств.

Обычно программная система Linux распространяется в виде *дистрибутива* (*distribution*), содержащего средства инсталляции и раскрутки системы. Большинству пользователей трудно самим собрать систему из разрозненных частей. Но не существует некоего стандартного дистрибутива - их много, каждый со своими преимуществами и недостатками. О различных дистрибутивах Linux речь пойдет в Разделе 2.1.

Несмотря на полноту Linux, вам все равно потребуется немножко ноу-хау по инсталляции и использованию UNIX как такового. Ни один из дистрибутивов Linux не свободен полностью от ошибок, так что вам может потребоваться что-то доделать "вручную" после инсталляции. Пользование UNIX - непростая задача, даже для коммерческих версий UNIX. Если у вас серьезные намерения относительно Linux, имейте в виду, что потребуются значительные усилия и внимание с вашей стороны, чтобы обеспечить нормальную его эксплуатацию. Это справедливо относительно *любого* UNIX, и Linux не является исключением. Из-за большой разношерстности Linux-сообщества и большого разнообразия в желаниях относительно возможностей системы не все и не для всех может быть быстро реализовано.

1.6.1 Замечания для новичков в UNIX

Инсталляция и использование вашего личного Linux не требует большой подготовки в UNIX как таковом. Действительно, многие новички в UNIX успешно инсталлируют Linux на своих компьютерах. Разумеется, желательны знания и опыт, но некоторых предварительные требования могут деморализовать. Если вам повезет, вы сможете инсталлировать и начать использовать Linux без каких-либо предварительных знаний по UNIX. Но как только вы столкнетесь с более сложными задачами эксплуатации Linux - инсталляцией новых программ, перекомпиляцией ядра и т.п., базовые знания по UNIX станут необходимыми.

К счастью, эксплуатируя свой Linux, вы имеете возможность освоить все существенные особенности UNIX, необходимые для выполнения этих задач. Эта книга содержит достаточно информации для начала работы - Глава 3 является учебным материалом по основам UNIX, а Глава 4 - по работе системного администратора Linux. Вы можете захотеть прочитать эти главы до попытки инсталляции Linux. Информация, содержащаяся там, может быть очень полезной, если вы столкнетесь с проблемами.

Никто не сможет превратиться из новичка в ОС UNIX за одну ночь в системного администратора UNIX. Вы должны быть готовы к длительному путешествию, которое вам предстоит.

1.6.2 Замечания для UNIX-гуру

Даже люди, имеющие длительный опыт работы с UNIX, могут нуждаться в дополнительных знаниях для инсталляции и использования Linux. Существуют аспекты системы, с которыми даже экспертам в UNIX нелишне познакомиться перед началом работы, хотя бы потому, что Linux - это не коммерческий UNIX. Он следовал, скорее всего, другим стандартам и принципам, чем те, к которым вы привыкли ранее. И хотя стабильность системы - важный фактор развития Linux, это *не единственный* фактор.

Возможно, более важной является функциональность. Во многих случаях новый код добавляется в ядро, даже если содержит ошибки и функционально не полон. Существует предположение, что важнее выпустить код, который пользователи могут тестировать и использовать, чем делать его недоступным. Например, WINE (Microsoft

Windows Emulator for Linux) имел "официальную" альфа версию до того, как был полностью оттестирован. Linux-сообщество имеет большой шанс поработать с кодом, протестировать его и помочь в его совершенствовании, пока те, кто считают альфа-версию достаточно хорошей для своих нужд, могут ее использовать. Поставщики коммерческих UNIX редко распространяют их таким образом.

Если вы уже более десятилетия системный администратор UNIX и работали с коммерческими версиями UNIX, Linux может потребовать времени на привыкание. Эта система современная и динамичная. Новые версии ядра появляются примерно раз в несколько недель. Постоянно появляются новые программы. Еще сегодня ваша система может быть совершенно современной, а уже завтра - каменным веком.

Учитывая такую динамику, как удержаться на плаву в мире Linux? Для большинства лучше всего делать частичные усовершенствования, то есть менять только те части, которые, на ваш взгляд *действительно нуждаются* в обновлении. Например, если вы не используете Emacs, нет смысла регулярно менять его версии на вашем компьютере. Более того, даже если вы приверженец Emacs, не нужно менять версию, если только в новой версии не появились те возможности, которые вам действительно необходимы. Совсем не обязательно всегда иметь самую последнюю версию всего.

Мы надеемся, что Linux оправдает или даже превзойдет ваши ожидания, связанные с "доморощенной" UNIX-системой. В самой основе Linux дух свободного программирования, постоянного развития и совершенствования. Linux-сообщество предпочитает экспансию стабильности, и эту концепцию трудно проглотить многим, особенно тем, кто стоит высоко в мире коммерческого UNIX. Нельзя считать Linux совершенством, как и ничто в мире свободно распространяемых программ. Но мы верим, что Linux полон и работоспособен, как и любая другая реализация UNIX.

1.7 Различия между Linux и другими операционными системами

Важно понимать различия между Linux и другими операционными системами, такими как MS-DOS, OS/2, а также другими реализациями UNIX для персональных компьютеров. Прежде всего, должно быть ясно, что Linux может счастливо сосуществовать с другими операционными системами на той же машине. Как мы увидим, существуют даже способы взаимодействия операционных систем.

1.7.1 Почему Linux?

Почему стоит использовать Linux вместо хорошо известных, хорошо оттестированных, хорошо документированных коммерческих операционных систем? Мы можем привести тысячи причин. Одна из наиболее важных - то что Linux - отличный выбор для персональных вычислений в среде UNIX. Если вы разработчик программ в UNIX, зачем дома использовать MS-DOS? Linux позволит вам создавать и тестировать программы для UNIX на вашем персональном компьютере, включая базы данных и приложения для X Window. Если вы студент, то высока вероятность, что университетская компьютерная система работает под UNIX. Linux позволяет вам иметь свой собственный UNIX и перекраивать его по своему вкусу. Инсталляция и использование Linux - также прекрасный путь изучения UNIX, если у вас нет доступа к другим UNIX-машинам.

Но не будем зарываться. Linux не для отдельных скромных любителей UNIX на персоналках. Это большая и достаточно сложная система для решения сложных задач и организации распределенных вычислений. Многие фирмы, особенно небольшие, двигаются в сторону Linux, предпочитая его другим UNIX. Университеты считают Linux отлично подходящим для обучения операционным системам. Крупные поставщики программ начинают понимать, какие выгоды сулит свободное распространение операционных систем.

Следующий раздел покажет наиболее важные различия между Linux и другими операционными системами. Мы надеемся, что Linux может удовлетворить все ваши вычислительные потребности или (в крайнем случае) значительно расширит возможности вашей вычислительной среды. Имейте в виду, что лучший способ узнать вкус Linux - это на нем поработать. Вам не обязательно даже инсталлировать полную систему, чтобы почувствовать его. Это мы покажем в Главе 2.

1.7.2 Linux против MS-DOS

Не является чем-то экзотическим одновременно держать на компьютере Linux и MS-DOS. Многие пользователи Linux работают с прикладными пакетами MS-DOS, вроде различных редакторов.

Хотя Linux имеет собственные аналоги для таких приложений (например, TeX), существуют разнообразные причины, по которым конкретный человек будет использовать MS-DOS наряду с Linux. Если вся ваша диссертация написана с использованием WordPerfect в MS-DOS, возможно вы не сможете конвертировать его в TeX или какой-то другой формат. Существует много коммерческих приложений для MS-DOS, которых нет в Linux, поэтому, если есть смысл, почему не использовать обе операционные системы.

Как вы, очевидно, знаете, MS-DOS не использует полностью функциональные возможности 80386 и 80486 процессоров. С другой стороны, Linux полностью работает в защищенном режиме процессора и реализует все возможности процессора. Вы можете иметь прямой доступ ко всей имеющейся в распоряжении памяти (и сверх того - используя виртуальную RAM). Linux обеспечивает полный UNIX-интерфейс, отсутствующий в MS-DOS. На Linux вы можете просто писать и отлаживать прикладные программы для UNIX, в то время, как это несложно делать под MS-DOS.

Мы можем обсуждать плюсы и минусы MS-DOS и Linux бесконечно. Между тем, давайте заметим, что Linux и MS-DOS абсолютно разные системы. MS-DOS - это дешевая ОС (в сравнении с другими операционными системами), и имеет широкую поддержку в мире персональных компьютеров. Ни одна другая ОС для персональных компьютеров не может сравниться с ней по популярности, прежде всего из-за стоимости. Мало кто из владельцев персональных компьютеров может даже представить, что однажды он потратит \$1000 или более только на одну операционную систему. Linux же, кстати, вообще бесплатен, так что у вас есть повод подумать.

Вы можете сами составить свое впечатление от сравнения Linux и MS-DOS, основываясь на том, насколько они отвечают вашим ожиданиям. Linux - это не для всякого. Если вам всегда хотелось иметь полномасштабный UNIX дома, не совершая больших затрат, Linux может быть как раз то, что вам надо.

Существует инструментарий, позволяющий взаимодействовать Linux и MS-DOS. Например, просто получить доступ к файлам MS-DOS из Linux. Есть также эмулятор MS-DOS, позволяющий выполнять многие популярные прикладные пакеты MS-DOS. Эмулятор Microsoft Windows находится на этапе создания.

1.7.3 Linux против прочих

Ряд других продвинутых операционных систем всходит на горизонте мира персональных компьютеров. В частности, OS/2 фирмы IBM и Windows NT фирмы Microsoft становятся все более популярны по мере ухода пользователей из MS-DOS.

Обе OS/2 и Windows NT являются полными многозадачными операционными системами, как и Linux. Чисто технически, OS/2, Windows NT и Linux очень похожи: они имеют похожие интерфейсы с пользователем, систему защиты и т.п. Но главное действительное отличие состоит в том, что Linux есть разновидность UNIX, а отсюда все преимущества принадлежности к UNIX-сообществу.

Что делает UNIX столь важным? Это не только самая популярная операционная система для многопользовательских машин, это также база для большей части свободно распространяемых в мире программ. Если у вас есть доступ к Internet, почти все программы, свободно доступные там, написаны именно для UNIX. (Сам Internet в большой степени стоит на UNIX).

Существует много различных фирм, производящих UNIX и ни одной, ответственной за его распространение. Существует большая тяга к стандартизации в UNIX-сообществе, которая выражается в концепции открытых систем. Но ни одна фирма не контролирует этот процесс. Поэтому любой производитель (или как показала практика - хакер) может следовать стандартам UNIX (коль скоро на них нет авторских прав).

OS/2 и Windows NT принадлежат частным компаниям. Поэтому интерфейс и проектные решения контролируются конкретными фирмами и только эти фирмы могут совершенствовать свои продукты. (Не надейтесь увидеть когда-нибудь в обозримом будущем бесплатную версию OS/2). В некотором смысле такая организация дела имеет преимущества: она обеспечивает жесткую стандартизацию программного и пользовательского интерфейсов. OS/2 есть OS/2, где бы вы ее не обнаружили, то же самое с Windows NT.

А интерфейс UNIX постоянно совершенствуется и меняется. Несколько организаций пытаются выработать стандарт программной модели, но эта задача очень сложная. Linux наилучшим образом соответствует стандарту POSIX.1 для программного интерфейса UNIX. Предполагалось, что Linux примкнет еще к ряду движений по стандартизации, но это не стало пока в число главных задач Linux-сообщества.

1.7.4 Другие реализации UNIX

Существует ряд других реализаций UNIX для 80386 и 80486. Архитектура 80386 сама подталкивает к проектированию UNIX, поэтому многие разработчики воспользовались этим преимуществом. Другие реализации UNIX, учитывавшие особенности архитектуры процессора весьма похожи на Linux. Вы можете убедиться, что почти все коммерческие версии UNIX поддерживают практически одинаковую программную среду и сетевые характеристики. Однако имеются и значительные отличия между Linux и коммерческими UNIX.

Прежде всего Linux поддерживает иной спектр аппаратных средств. Linux поддерживает большинство хорошо известных устройств, но поддержка ограничена той аппаратурой, к которой пользователи действительно имеют доступ. Разработчики коммерческих UNIX обычно имеют больший список устройств, хотя Linux и отстает незначительно. Про особенности аппаратуры поговорим в Разделе 1.8.

Во-вторых, коммерческие реализации UNIX обычно идут в связке с полным набором документации, а также с обязательствами разработчика по сопровождению. Документация на Linux ограничивается той, которую можно найти в Internet, да еще книжки вроде этой. В Разделе 1.9 мы перечислим документацию и другие источники информации.

Коль скоро речь идет о стабильности и надежности, многие пользователи отмечают, что Linux по крайней мере не менее надежен, чем коммерческие UNIX. Linux все еще находится в стадии развития, и некоторые вещи (вроде TCP/IP) недостаточно стабильны, но постоянно совершенствуются.

Наиболее важным фактором для многих является цена. Linux распространяется свободно, если вы имеете доступ в Internet (или другую компьютерную сеть) и можете его скачать. Если у вас нет выхода в такую сеть, то вы можете купить его на дискетах или CD-ROM (смотрите Приложение B). Разумеется, вы можете скопировать Linux у друга или разделить с кем-то стоимость его покупки. Если вы планируете инсталлировать Linux на большом количестве машин, вам достаточно купить всего одну копию. На тиражирование нет лицензионных ограничений.

Не следует преуменьшать реальную стоимость коммерческих UNIX, поскольку наряду со стоимостью собственно программ, сюда входят также стоимость документации, сопровождение, гарантия качества. Это очень важные составляющие для больших организаций, но, может быть, не столь существенные для индивидуальных пользователей. В любом случае, во многих фирмах и в университетах считают, что использование Linux в лабораториях на недорогих персональных компьютерах предпочтительнее коммерческого UNIX в лаборатории, укомплектованной рабочими станциями.

Как пример из "реального мира", можно сказать о том, что Linux путешествовал по северу Тихого океана, выполняя работу по телекоммуникации и анализу данных на океанографическом исследовательском судне. Linux используется на исследовательской станции в Антарктике. Несколько госпиталей используют Linux для ведения историй болезни. Он доказал, что он надежен и удобен, как и другие реализации UNIX.

Существуют и другие бесплатные или недорогие реализации UNIX для 386 и 486. Одна из наиболее известных реализаций 386BSD. 386BSD совместима с Linux во многих аспектах, но какая из них "лучше" зависит от ваших личных желаний и ожиданий. Мы можем указать одно существенное отличие: Linux создавался открыто (когда каждый доброволец мог внести свой вклад в процесс создания), а 386BSD создан замкнутой группой программистов, которые и поддерживают систему. Поэтому существует заметное различие в философии и разработке между этими двумя проектами. Цели двух проектов существенно различны: цель Linux - создать полную UNIX-систему от начала (и получить большое удовольствие от этого процесса); а цель 386BSD частично состоит в модификации существующего кода BSD, применительно к 386.

NetBSD - это другая версия BSD NET/2 для нескольких машин, включая 386. NetBSD имеет слегка более открытую концепцию разработки и сравнима с 386BSD по многим аспектам.

Другой проект HURD - попытка Free Software Foundation создать и распространять свободную версию UNIX для многих платформ. За дополнительной информацией об этом проекте обращайтесь в Free Software Foundation (адрес дан в Приложении A). В момент написания книги проект HURD все еще на начальной стадии.

Существуют также другие недорогие версии UNIX, например Coherent (приблизительно $99) и Minix (академический, но полезный UNIX, на котором первоначально базировался Linux). Некоторые из этих реализаций представляют преимущественно академический интерес, в то время, как другие - нормальные полномасштабные системы. Нет смысла говорить о том, как много индивидуальных пользователей UNIX двигаются в сторону Linux.

1.8 Требования к оборудованию

Вы должны быть убеждены, что Linux прекрасен и что вы пополните его грандиозными вещами. Но это потом, а до того, как броситесь его инсталлировать, вы должны сориентироваться в требованиях к аппаратуре, которые диктует Linux.

Имейте в виду, что Linux был создан самими пользователями. Это означает, что большая часть поддерживаемого Linux оборудования - это то, что пользователи реально у себя имеют. Как в результате оказалось - большая часть популярной периферии для 80386/80486 поддерживается (действительно, Linux поддерживает оборудование, которое в ряде случаев не поддерживают некоторые коммерческие UNIX). Хотя некоторые достаточно экзотические устройства пока не поддерживаются. Если какое-то из облюбованных вами устройств пока не поддерживается в Linux, есть смысл надеяться, что оно скоро будет поддерживаться.

Многие компании лицензируют интерфейс устройств, поэтому добровольные разработчики Linux просто не могут написать эти драйверы. Иначе это будет нарушением авторских прав соответствующей компании.

Мало что можно изменить в этой ситуации. Иногда хакеры пишут драйверы, основываясь на своих представлениях о конкретном интерфейсе. В других случаях разработчики с разной степенью успеха пытаются работать с соответствующими компаниями, пытаясь получить информацию об интерфейсе.

В следующем разделе мы попытаемся дать резюме технических требований Linux. *Linux Hardware HOWTO* (смотрите Раздел 1.9) содержит более полный перечень оборудования, поддерживаемого Linux.

Большая часть драйверов Linux в настоящее время находится в стадии разработки. Различные дистрибутивы могут содержать разные наборы драйверов. Здесь прежде всего перечисляются те драйверы, которые уже поддерживаются определенное время и зарекомендовали себя как достаточно стабильные. (Дополнительную информацию по дистрибутивам смотрите в Разделе 2.1).

1.8.1 Требования к материнской плате и процессору

В настоящее время Linux поддерживает системы на Intel 80386, 80486 или Pentium CPU. Это включает все вариации этих процессоров, такие как 386SX, 486SX, 486DX и 486DX2. С Linux могут работать также "неинтеловские" клоны процессоров, вроде AMD и Cyrix.

Если у вас 80386 или 80486SX, вы можете также иметь сопроцессор, хотя это и не обязательно, (ядро Linux может эмулировать FPU). Поддерживаются все стандартные сочетания FPU, такие сопроцессоры как IIT, Cyrix FasMath и Intel.

Материнская плата должна использовать шину ISA или EISA. Это определяет, как система взаимодействует с периферией и другими компонентами главной шины. Большинство продаваемых сегодня систем имеют ISA или EISA. Шина MicroChannel (MCA) фирмы IBM, используемая на машинах типа IBM PS/2, пока не поддерживается.

Поддерживаются системы с локальными шинами, ускоряющими доступ к видео и дискам. Предполагается, что у вас стандартная локальная шина, вроде VESA Local Bus (VLB).

1.8.2 Требования к памяти

Linux требует совсем немного памяти в сравнении с другими развитыми операционными системами. Вы должны иметь как минимум 2Мбайт RAM; хотя настоятельно рекомендуется иметь не менее 4 Мбайт. Чем больше памяти - тем быстрее работает система.

Linux может поддерживать все 32-битовое адресное пространство процессоров 386/486; другими словами, он автоматически использует всю память.

Linux может успешно работать на 4 Мбайтах RAM, включая всяческие свистульки и погремушки, вроде X Window, Emacs и т.п. Хотя, иметь побольше памяти не менее важно, чем иметь помощнее процессор. 8 Мбайт более подходит для индивидуального использования; 16 Мбайт или более - если вы предполагаете более серьезную загрузку системы.

Большинство пользователей Linux выделяют часть жесткого диска для области своппинга, которая используется как виртуальная RAM. Если даже вы имеете много реальной физической памяти, RAM, вы можете пожелать иметь область своппинга. Хотя область своппинга не заменяет действительной физической памяти, она может позволить выполнять на вашей системе более об'емные приложения, удаляя неактивную часть программы на диск. Размер области своппинга, которую вы должны выделить, зависит от нескольких факторов; мы вернемся к этому в Разделе 2.2.3.

1.8.3 Требования к драйверам жесткого диска

Вам не обязательно иметь драйвер жесткого диска для работы в Linux. Вы можете работать с минимальной системой с гибкого диска. Но это медленно и имеет много ограничений, да и большинство пользователей имеет доступ к памяти на жестких дисках. У вас должен быть 16-ти битный контроллер в стандарте AT. В ядре есть поддержка 8-ми битного XT-стандарта; хотя разумеется, большинство контроллеров использует сегодня AT-стандарт. Linux может поддерживать все MFM, RLL и IDE контроллеры. Поддерживается большинство (но не все) ESDI контроллеры - только те, которые обеспечивают эмуляцию ST506.

Общее правило для не-SCSI драйверов жестких дисков состоит в том, что если вы можете иметь доступ к этим устройствам из MS-DOS или другой ОС, значит вы можете работать с ними и в Linux.

Linux может также поддерживать многие популярные SCSI контроллеры, хотя поддержка SCSI ограничена из-за большого разнообразия существующих стандартов таких интерфейсов. Поддержка SCSI контроллеров включает Adaptec AHA1542B, AHA1542C, AHA1742A (BIOS version 1.34), AHA1522, AHA1740, AHA1740 (SCSI-2 controller, BIOS 1.34 in Enhanced mode); Future Domain 1680, TMC-850, TMC-950; Seagate ST-02; UltraStor SCSI; Western Digital WD7000FASST. Также должны работать клоны, базирующиеся на этих картах.

1.8.4 Требования к дисковому пространству

Разумеется, для инсталляции Linux вам необходимо иметь некоторое свободное пространство на жестком диске. Linux поддерживает различные драйверы жестких дисков на одной машине; вы можете выделить место для нескольких устройств, если это необходимо.

Размер необходимого пространства зависит в большой степени от ваших потребностей и программ, которые вы инсталлируете. Linux сравнительно компактный UNIX; вы можете для всей системы занять 10 - 20 Мбайт. Между тем, если вы хотите иметь место для расширения и для больших пакетов, вроде X Window, вам потребуется больше места. Если вы планируете множественный доступ, вам потребуется иметь место и для файлов других пользователей.

Кроме того, даже если вы имеете большое количество физической RAM (16 Мбайт или более), скорее всего вы захотите иметь область своппинга, используемую виртуальной памятью. Детали мы обсудим в Разделе 2.2.3.

Каждый дистрибутив Linux обычно сопровождается какой-то литературой, которая может помочь вам определиться с об'емом необходимой памяти в зависимости от того, какое программное обеспечение вы планируете поставить. Минимальную систему вы можете эксплуатировать менее, чем на 20 Мбайтах. Полная система со всеми свистульками и погремушками потребует до 80 Мбайт. Очень большие системы для многих пользователей, где зарезервировано место для последующих расширений, потребует 100 - 150 Мбайт. Но это лишь грубые прикидки, которые вы уточните исходя из своих потребностей.

1.8.5 Требования к монитору и видеоадаптеру

Linux поддерживает все стандарты Hercules, CGA, EGA, VGA, IBM monochrome, and Super VGA видео карт и текстовые мониторы. В общем случае, если видеокарта и монитор работают под другими ОС вроде MS-DOS, они будут работать и под Linux. Оригинальные карты IBM CGA дают в Linux "снег", так что их неприятно использовать.

Графическое окружение вроде X Window имеет свои требования к видеооборудованию. Вместо перечисления этих требований мы перенесем обсуждение в Раздел 5.1.1. А кратко, для работы с X Window System на вашем Linux, вам требуется одна из видеокарт, перечисленных в этом разделе.

1.8.6 Прочее оборудование

У большинства пользователей есть "специфическое" оборудование, вроде стриммера, памяти на CD-ROM, саунд карты и т.д., поэтому они интересуются, поддерживается ли оно в Linux. Читайте дальше.

Мышь и другие устройства, подключаемые к портам Большей частью мышь вы будете использовать в графическом окружении, таком как X Window System. Но некоторые приложения Linux, не ассоциируемые с графической средой, также используют мышь.

Linux поддерживает все стандарты последовательно подключенной мыши, включая Logitech, серию MM, Mouseman, Microsoft (2-кнопки) и Mouse Systems (3-кнопки). Linux также поддерживает мышь, подключенную на шину: Microsoft, Logitech и ATIXL. Поддерживается также интерфейс мыши PS/2.

Все прочие подключаемые таким же образом устройства, которые эмулируют вышеперечисленных мышей, тоже должны работать.

Память на CD-ROM Почти все драйверы CD-ROM используют интерфейс SCSI. Если у вас есть SCSI-адаптер, поддерживаемый Linux, то ваш CD-ROM должен работать. Проверена работоспособность ряда драйверов для CD-ROM под Linux, включая NEC CDR-74, Sony CDU-541 и Texel DM-3024. Linux поддерживает также драйверы Sony internal CDU-31a и Mistsumi CD-ROM.

Linux поддерживает также стандарт ISO-9660 файловой системы для CD-ROM.

Драйверы стриммеров Сейчас на рынке имеется ряд стриммеров. Большинство из них используют SCSI-интерфейс и практически все поддерживаются Linux. Среди проверенных устройств Sankyo CP150SE; Tandberg 3600; Wangtek 5525ES, 5150ES и 5099EN с адаптером PC36. Другие QIC-02 устройства также должны поддерживаться.

Идет постоянная разработка драйверов для различных новых устройств, таких как Colorado, который вешается на контроллер гибкого диска.

Принтеры Linux поддерживает весь спектр параллельных принтеров. Если вы можете подключить ваш принтер на параллельный порт в MS-DOS или в другой операционной системе, то он может работать и в Linux.

Программная поддержка принтера в Linux состоит из стандартных для UNIX программ lp и lpr. Эти программы позволяют также организовать удаленную печать через сеть.

Модемы Как и с поддержкой принтеров, Linux поддерживает полный спектр последовательных модемов, как внешних, так и внутренних. В Linux много программ телекоммуникации, включая Kermit, pcomm, minicom и Seyon. Если ваш модем может работать с другой операционной системой, вы сможете с ним работать и в Linux без каких-либо проблем.

Карты Ethernet Многие популярные карты Ethernet и LAN-адаптеры поддерживаются в Linux. Они включают:

- 3com 3c503, 3c503/16
- Novell NE1000, NE2000
- Western Digital WD8003, WD8013
- Hewlett Packard HP27245, HP27247, HP27250
- D-Link DE-600

Есть сведения о работе следующих клонов:

- LANNET LEC-45
- Alta Combo
- Artisoft LANtastic AE-2
- Asante Etherpak 2001/2003,
- D-Link Ethernet II
- LTC E-NET/16 P/N 8300-200-002
- Network Solutions HE-203,
- SVEC 4 Dimension Ethernet
- 4-Dimension FD0490 EtherBoard 16

Должны также работать и любые другие клоны, совместимые с перечисленными выше.

1.9 Источники информации по Linux

Как вы, возможно, догадались, существует много источников информации по Linux помимо этой книги. В частности, есть ряд книг, не конкретно по Linux, а скорее по UNIX вообще, которые могут очень помочь, особенно тем, кто не имеет предварительного опыта в UNIX. Если вы не знакомы с миром UNIX, мы настоятельно советуем вам уделить время одной из таких книг, прежде чем храбро ринуться в джунгли Linux. В качестве хорошего начала может быть использована книга Grace Todino и John *Learning the UNIX Operating System*.

Многие из нижеупомянутых источников доступны он-лайн в электронном виде. То есть для того, чтобы ими воспользоваться, вы должны иметь доступ к Internet, USENET или Fidonet. Если у вас нет он-лайн доступа к этим материалам, то следует найти добряка, который сделает вам твердую копию.

1.9.1 Документация, доступная по он-лайн

Если у вас есть доступ к Internet, вы можете найти много документации по Linux через anonymous FTP из архивов, расположенных по всему миру. Если же вы не имеете прямого доступа к Internet, эту документацию все-таки можно достать: много дистрибутивов Linux, содержащих документы, упомянутые здесь, продается на CD-ROM. Они также распространяются по многим другим сетям, вроде Fidonet и CompuServe. Если вы в состоянии послать почту в Internet, значит вы можете достать и эти файлы, используя один из ftpmail-серверов, которые пошлют вам документы или файлы электронной почтой из FTP-архивов. Смотри в Приложении C дополнительную информацию по использованию `ftpmail`.

Существует большое число FTP-архивов, содержащих тексты программ Linux и соответствующую документацию. Список известных архивов Linux дан в Приложении C. Для того, чтобы минимизировать сетевой трафик, всегда следует использовать FTP-сервер, находящийся в географической близости.

Приложение A содержит список документации на Linux, доступной через anonymous FTP. Имена файлов будут различаться на разных серверах. Большинство серверов хранит документацию на Linux в подкаталоге docs Linux-овского архивного подпространства. Например, на FTP сервере `sunsite.unc.edu` файлы Linux помещены в каталог `/pub/Linux` , а документация по Linux - в `/pub/Linux/docs`.

Примеры доступной документации - это *Linux FAQ* (Frequently Asked Questions), собрания часто задаваемых вопросов (в данном случае) по Linux; документация *Linux HOWTO*, описывающая специфические аспекты системы, включая *Installation HOWTO*, *Printing HOWTO*, *Ethernet HOWTO* и Linux META-FAQ - перечень других источников информации по Linux в Internet.

Большинство этих документов регулярно посылается в несколько относящихся к Linux групп USENET; смотрите Раздел 1.9.4.

1.9.2 Linux И WWW (World Wide Web)

Начальная страница (Home Page) Документации на Linux доступна на WWW по адресу `http://sunsite.unc.edu/mdw/linux.html`

Эта страница содержит много HOWTO и других документов в формате HTML, а также ссылки на другие сервера, интересные для пользователей Linux.

1.9.3 Книги и другие публикации

В настоящее время существует всего несколько публикаций специально посвященных Linux. Большей частью это книги из Linux Documentation Project (LDP) - проекта, реализуемого в Internet по написанию и распространению "руководств" по Linux. Эти руководства аналогичны документации, поставляемой с коммерческими версиями UNIX: они покрывают все - от инсталляции Linux до его использования, программирования, работы в сети, разработки ядра и т.д.

Руководства LDP доступны через FTP-серверы Internet, а также по обычной электронной почте из нескольких источников (см. Приложение A).

И хотя собственно по Linux книг немного, тем не менее есть большое число книг по UNIX вообще, которые, без сомнения, могут использоваться применительно к Linux, поскольку Linux не имеет значительных отличий от других реализаций UNIX. Короче, все, что вы хотите узнать о программировании на Linux, может быть найдено в этих книгах, предназначенных для широкой аудитории пользователей UNIX. А здесь мы представляем наиболее важные специфические детали Linux и надеемся, что вы также будете обращаться за деталями и к другим источникам.

Вооружившись некоторым количеством хороших книжек по UNIX, а также данной книгой, вы будете способны разобраться во всем. Приложение A включает список настоятельно рекомендуемых книг по UNIX, как для новичков в UNIX , так и для асов.

Есть также ежемесячный журнал по Linux, под названием *Linux Journal*. Он распространяется по всему миру и представляет хорошую возможность быть в курсе происходящего в Linux- сообществе, особенно, если вы не имеете доступа к новостям USENET (см. ниже). Относительно подписки на *Linux Journal* см. Приложение A.

1.9.4 Новости USENET

USENET - это мировые электронные новости и одновременно клубы по интересам - так называемые “newsgroups”(у нас принято говорить - "телеконференции"). Многое по развитию Linux делается через Internet и USENET, так что не удивляет существование ряда телеконференций USENET, посвященных Linux.

Первоначально была создана телеконференция `alt.os.linux`, чтобы вынести дискуссии по Linux из comp.os.minix. Но скоро трафик `alt.os.linux` вырос настолько, что в феврале 1992 года было проведено голосование о включении конференции в иерархию "comp" и создана телеконференция `comp.os.linux`.

`comp.os.linux` быстро стала одной из наиболее популярных (и шумных) телеконференций USENET, более популярной, чем другие группы иерархии `comp.os`. В декабре 1992, в результате голосования группа была разбита, чтобы уменьшить об'емы трафика, но только `comp.os.linux.announce` прошла тогда по голосованию. В июле 1993 , наконец-то группа была разбита на иерархию групп. За это проголосовало около 2000 человек - одно из самых больших голосований в истории USENET.

Если у вас нет прямого доступа к USENET, но имеется возможность посылать и получать письма по e-mail, то в Internet существуют шлюзы, через которые можно получать телеконференции:

comp.os.linux.announce

это модерируемая (контролируемая) телеконференция, содержащая об'явления и важные сообщения относительно Linux (например, сообщения об ошибках, существенные исправления и т.п.). Если вы вообще собираетесь читать телеконференции Linux - эту читайте обязательно. Часто важные сообщения этой конференции не дублируются в другие. Также в этой же телеконференции бывают важные оперативные сообщения.

Любое сообщение в этой группе должно быть одобрено модераторами Matt Welsh и Lars Wirzenius. Если вы хотите выставить свое сообщение в этой группе, вы обычным порядком посылаете его в телеконференцию. Оно автоматически будет направлено для одобрения модератору. Но если ваша система новостей не настроена, можно направить статью электронной почтой по адресу `linux-announce@tc.cornell.edu`. Все остальные телеконференции этой иерархии linux, перечисленные ниже - немодерируемые.

comp.os.linux.help

Это наиболее популярная телеконференция Linux. Она предназначена для вопросов и ответов по использованию, установке и другим вопросам эксплуатации Linux. Если у вас проблемы с Linux - пишите в эту группу и с большой вероятностью получите ответ от кого-нибудь, способного вам помочь. Но очень желательно ознакомиться со всеми доступными вам материалами, прежде, чем задавать вопрос.

comp.os.linux.admin

Здесь обсуждаются вопросы эксплуатации Linux (обычно в многопользовательском режиме) Обсуждается работа администратора системы Linux (например, сохранение системы, работа с пользователями и т.п.).

comp.os.linux.development

Эта телеконференция для обсуждения проблем развития Linux. Вопросы, касающиеся ядра, системных библиотек и системных программ. Например, если вы пишете драйвер и нуждаетесь в помощи по некоторым деталям программирования - как раз в эту телеконференцию следует посылать такие вопросы. Эта конференция предназначена также для обсуждения целей и направления развития Linux, как это описывалось в Разделе 1.6.

Следует заметить, что эта телеконференция для обсуждения проблем разработки *собственно* Linux. Прикладное программирование в Linux обсуждается в других телеконференциях.

comp.os.linux.misc

Эта телеконференция для обсуждения проблем, не подходящих для других групп. Особенно для "защитников" Linux (для статей типа "Linux против Windows NT"). Всякие нетехнические окололинуксовские споры могут происходить здесь.

Следует заметить, что телеконференция comp.os.linux была заменена иерархией телеконференций. Если у вас есть доступ к `comp.os.linux` и нет доступа к другим телеконференциям этой иерархии, вдохновите администратора их завести.

1.9.5 Списки рассылки Internet

Если у вас есть доступ к электронной почте Internet, вы можете включиться в списки рассылки, даже если вы не имеете доступа к USENET. Даже если вы не можете работать непосредственно в Internet, но можете обмениваться с ней почтой (например, UUCP, FidoNET, CompuServe и другие сети имеют почтовую связь с Internet), вы можете подключиться к какому-то списку рассылки. Список рассылки ``Linux Activists'' предназначен для разработчиков Linux и людей, заинтересованных в содействии процессу разработки. Это "многоканальная" рассылка; можно подключиться к разным ее ветвям: NORMAL - общие вопросы Linux; KERNEL - разработка ядра; GCC - разработка gcc-компилятора и библиотек; NET - протоколы TCP/IP; DOC - документация по Linux; и другие.

За дополнительной информацией о списках рассылки, связанных с Linux, пишите:

```
linux-activists@niksula.hut.fi
```

Вы сможете получить действующий перечень тем рассылки, включая информацию о том, как включиться в список рассылки и как отписаться.

Существует несколько списков рассылки для Linux. Лучше всего для ориентации посмотреть об'явления в телеконференциях Linux USENET, а также просмотреть списки тем рассылки, периодически публикуемые в USENET group `news.answers`.

1.10 Получение помощи

Без сомнения, вам понадобится какая-то помощь во время ваших приключений в мире Linux. Даже самые крутые из крутых юниксистов временами спотыкаются о какие-то закорючки Linux, поэтому важно знать, где и как искать помощи, когда потребуется.

Первоочередной источник помощи - списки рассылки и телеконференции USENET обсуждались в Разделе 1.9. Если у вас нет к ним онлайнового доступа, вы можете поискать доступ к такой информации на локальных BBS, в CompuServe и т.д.

Ряд фирм предлагают коммерческую поддержку Linux. Вы можете платить за "подписку", что позволит вам созваниваться с консультантами по поводу возникающих проблем с Linux. Но при наличии доступа к USENET и почте Internet вы можете найти и бесплатное сопровождение.

Но сначала ищите ответы во всей доступной вам документации! Прежде всего в источниках, перечисленных в Разделе 1.9 и Приложении A. Эти документы тщательно написаны для людей вроде вас, которые нуждаются в помощи при работе с Linux. Даже книги, написанные про UNIX вообще, применимы к Linux и вы должны извлекать из них пользу для себя, и как правило, найдете ответы на интересующие вас вопросы.

Приучайтесь к самостоятельности. В большинстве случаев предпочтительно самостоятельно исследовать проблему насколько возможно, прежде чем обращаться за помошью. Прежде всего обращайтесь за помошью к самому Linux. Помните, что Linux некоммерческая ОС и никогда не пыталась таковой выглядеть. Вы не умрете, если сами займетсь хакерством. Это приучит вас самостоятельно решать проблемы, может через некоторое время вы сами почувствуете себя в Linux гуру.

Сохраняйте спокойствие. Ни в коем случае не приходите в отчаяние в результате общения с системой. Вы ничего не доб'етесь от системы с помощью топора или, того лучше, с помощью мощного электромагнита. Linux взрослеет и мужает, становится надежнее, так что мы надеемся, что число проблем будет уменьшаться. Кстати, всяких фокусов можно ожидать и от коммерческих UNIX-ов.

Воздерживайтесь от скоропалительных решений. Многие люди бросаются рассылать письма по конференциям до того, как спокойно подумают. Часто решение находят через пять минут после того, как закончили взывать о помощи к мировому сообществу.

Если вы решили послать просьбу о помощи, постарайтесь это сделать наилучшим образом. Постарайтесь оформить просьбу максимально вежливо и предельно информативно. Не забывайте, что помощь в сети оказывается добровольно.

2 Приобретение и инсталляция Linux

В этой главе мы опишем, как приобрести Linux в виде подготовленных дистрибутивов и как их инсталлировать.

Мы уже говорили, что не существует "официального" дистрибутива Linux. Существует много дистрибутивов, каждый из которых хорош для определенных целей. Эти дистрибутивы можно взять с ftp-серверов Internet, с BBS, на дискетах и CD-ROM.

Здесь мы дадим общую картину процесса инсталляции. Каждый дистрибутив имеет свои конкретные инструкции по инсталляции, но вооруженные представленной здесь общей концепцией, вы сможете хорошо ориентироваться в существе этого процесса. Приложение A дает список источников информации, к которым можно обратиться, если вы окончательно запутаетесь.

Эта книга имеет дополнительные разделы, связанные с деталями Slackware-дистрибуции Linux.

2.1 Дистрибуции Linux

Поскольку Linux относится к свободно распространяемым программам, не существует организации или личности, отвечающей за его сопровождение. Поэтому каждый волен формировать и распространять дистрибутивы Linux, находясь в рамках GPL (General Public License). В результате, множество вариантов дистрибутивов Linux можно взять на FTP-серверах или по электронной почте.

Поэтому вы сталкиваетесь с проблемой, какой дистрибутив Linux вам нужен. Дистрибутивы бывают разные. Есть "всеоб'емлющие" дистрибутивы, содержащие все, что может использоваться в системе. Другие дистрибутивы Linux "маленькие", предназначены для пользователей, не обладающих неисчерпаемыми запасами дискового пространства. Многие дистрибутивы содержат только базовые программы Linux и предполагают в дальнейшем самостоятельную инсталляцию больших пакетов, вроде X Window System. (В Главе 4 мы опишем, как это делается).

В Приложении A содержится список дистрибутивов Linux, доступных по Internet и электронной почте.

Как выбрать из возможных дистрибутивов? Если у вас есть доступ к новостям USENET или другой системе компьютерных телеконференций, вы можете спросить мнение тех людей в сети, кто уже имеет опыт инсталляции Linux. Еще лучше - попросить помощи у знающего Linux, кого, в свою очередь, знаете вы. Надо учитывать много факторов при выборе дистрибутива, но мнения различных людей на этот счет различны. Но на самом деле большинство популярных дистрибутивов Linux содержат примерно один и тот же набор программ, так что можно выбирать, не очень мучаясь. В этой книге обсуждаются инсталляции популярных дистрибутивов Linux Slackware и Slackware Pro.

2.1.1 Получение Linux из Internet

Если у вас есть доступ к Internet, простейший способ получить Linux - это по FTP.[6] В приложении C дан список некоторых FTP-серверов с архивами, содержащими Linux. Один из них - `sunsite.unc.edu`, где различные дистрибутивы могут быть найдены в каталоге

```
/pub/Linux/distributions
```

Многие дистрибутивы хранятся в виде образов дискет. То есть дистрибутив состоит из множества файлов, каждый из которых содержит двоичный образ дискеты. Для того, чтобы скопировать содержимое на дискету, вы должны использовать программу `RAWRITE.EXE` под MS-DOS. Эта программа копирует поблочно содержимое файла на дискету, не анализируя формат диска.[7]

Программу `RAWRITE.EXE` можно взять на многих Linux FTP-серверах, включая `sunsite.unc.edu`, в каталоге

```
/pub/Linux/system/Install/rawrite
```

Во многих случаях вы просто считываете множество образов дискет и используете `RAWRITE.EXE` для каждого образа, чтобы создать дискету. Вы загружаетесь с так называемой "boot"-дискеты и можете далее скидывать систему. Как правило,

---

[6] Если вы не имеете прямого доступа к Internet, вы можете получить Linux через сервис `ftpmail` по электронной почте. См. приложение C.

[7] Если у вас есть доступ к UNIX-станции с дисководом, вы можете также использовать команду `dd` для копирования образа файла прямо на дискету. Команда, вроде `dd of=/dev/rfd0 if=foo bs=18k´´` осуществит прямое копирование содержимого файла `foo` на дискету на Sun-станции. Проконсультируйтесь у местных спецов по UNIX относительно использования команды `dd`.

программы инсталлируются прямо с дискет, хотя некоторые дистрибутивы позволяют инсталлировать из разделов MS-DOS винчестера. Некоторые дистрибутивы позволяют инсталлировать по сети, посредством TCP/IP. Документация на каждый дистрибутив должна описывать допустимые методы инсталляции.

Другие дистрибутивы Linux инсталлируются с дискет в формате MS-DOS. Например, дистрибутив Linux Slackware требует создания с помощью `RAWRITE.EXE` только дискет boot и root. Остальные дискеты копируются на дискеты в формате MS-DOS командой MS-DOS COPY. Система инсталлируется прямо с дискет MS-DOS. Это избавляет вас от необходимости долго пользоваться `RAWRITE.EXE`, но предполагает доступность MS-DOS для создания дискет.

Каждый дистрибутив, доступный по FTP, должен содержать файл `README`, описывающий, как скидывать на дискеты и как их готовить для инсталляции. Обязательно прочитайте всю доступную документацию относительно вашей версии.

При перекачке Linux необходимо использовать режим перекачки двоичных файлов (для большинства FTP-клиентов это команда "`binary`").

В Разделе 2.1.4 описывается получение дистрибутива Slackware по Internet.

2.1.2 Получение Linux из других он-лайн источников

Если вы имеете доступ к другим сетям, вроде CompuServe или Prodigy, то в них могут быть свои средства перекачки файлов. Кроме того, существует множество BBS, на которых есть Linux. Список Linux BBS дан в Приложении D. Не все дистрибутивы Linux доступны через эти сети, многие дистрибутивы на CD-ROM распространяются только почтой.

2.1.3 Получение Linux по почте

Если у вас нет доступа к Internet или BBS, множество дистрибутивов Linux можно получить по почте на дискетах, лентах, лазерных дисках. В Приложении B дан список таких дистрибуторов. Многие из них принимают оплату по кредитным карточкам и платежным поручениям, так что если вы живете даже не в США или Канаде, вы все равно сможете таким образом приобрести дистрибутив Linux.

Linux относится к свободно распространяемым программам, хотя его распространение и происходит под лицензией GPL. Поэтому пересылка Linux по почте может стоить вам US30 $- US$150, в зависимости от варианта дистрибутива. Однако, если вы знаете кого-либо, кто уже приобрел и скинул себе Linux, вы можете взять взаймы или скопировать его. Дистрибуторам Linux запрещено накладывать ограничения на его распространение в любой форме. Даже если вы намереваетесь инсталлировать Linux на все компьютеры вашей лаборатории, вам достаточно купить один экземпляр дистрибутива у любого из дистрибуторов.

2.1.4 Получение Slackware

Slackware - популярная дистрибуция Linux, сопровождаемая Patrick Volkerding.[8]

Эта версия легко инсталлируется и достаточно полна. Она может быть получена по Internet, а также на CD-ROM от ряда поставщиков (см. Приложение B).

Дистрибуция Slackware состоит из "наборов дисков" (“disk sets”), каждый из которых содержит конкретный тип программ (например, набор "d" содержит средства разработки *developmenttools*, такие как gcc-компилятор, и набор "x", содержащий X Window System). Вы можете выбирать, какие наборы инсталлировать; другие наборы вы можете инсталлировать позже.

Версия Slackware, описанная здесь - это версия 2.0.0, от 25 июня 1994. Инсталляция более поздних версий должна быть во многом аналогична.

Наборы дисков Slackware К сожалению, Slackware не имеет полного списка потребностей памяти для каждого набора. Вам потребуется не менее 7 Мбайт для инсталляции лишь серии "A"; очень грубая оценка необходимого дискового пространства - от 2 до 2.5 Мбайт на дискету.

Имеются следующие наборы дисков:

A

Базовая система. Достаточная для начала работы и делает доступными команды `elvis` и `comm` (`elvis` - текстовый редактор, comm - сравнение файлов). Базируется на ядре 1.0.9 и новом стандарте файловой системы (FSSTND). Известно, что эти диски хорошо размещать на дискетах 1.2M (что не характерно для остальных частей Slackware). Если у вас в распоряжении только дисковод на 1.2M, вы все равно можете инсталлировать базовую систему, сбросив другие диски и инсталлировать их с жесткого диска.

AP

Различные приложения и дополнения, вроде руководства, `groff`, `ispell` (GNU and international versions), `term`, `joe`, `jove`, `ghostscript`, `sc`, `bc`, и quota patches.

---

[8] С Patrick Volkerding можно связаться по Internet: volkerdi@mhd1.moorhead.msus.edu.

D

Разработка программ. GCC/G++/Objective C 2.5.8, `make` (GNU и BSD), `byacc` и GNU `bison`, `flex`, the 4.5.26 C libraries, `gdb`, kernel source for 1.0.9, `SVGAlib`, `ncurses`, `clisp`, `f2c`, `p2c`, `m4`, `perl`, `rcs`.

E

GNU Emacs 19.25.

F

Набор Часто Задаваемых Вопросов (`FAQ`s) и другая документация.

I

Info-страницы для GNU-программ. Документация для различных программ, читаемая с помощью info или Emacs.

N

Сетевые программы. TCP/IP, UUCP, `mailx`, `dip`, `deliver`, `elm`, `pine`, `smail`, `cnews`, `nn`, `tin`, `trn`.

OOP

Об'ектно-Ориентированное Программирование. GNU Smalltalk 1.1.1 и Smalltalk Interface to X (STIX).

Q

Исходники Alpha-ядра (во время написания книги - Linux 1.1.18).

TCL

Tcl, Tk, TclX, blt, itcl.

Y

Игры. Набор игр BSD и Tetris для терминалов.

X

Базовая система XFree86 2.1.1 с libXpm, fvwm 1.20 и xlock.

XAP

X-приложения: X11 `ghostscript`, `libgr13`, `seyon`, `workman`, `xfilemanager`, xv 3.01, GNU `chess` и `xboard`, `xfm` 1.2, `ghostview`, и различные X-игры.

XD

Программы работы в X11. X11 библиотеки, сервер linkkit, поддержка PEX.

XV

Xview 3.2 release 5. XView библиотеки, виртуальный и невиртуальный Open Look window-менеджеры.

IV

Interviews библиотеки, include-файлы, doc и `idraw` приложения.

OI

ParcPlace's Object Builder 2.0 и Object Interface Library 4.0. Обратите внимание, что она работает только с libc-4.4.4, но будет новая версия, как только станет доступным gcc 2.5.9.

T

Системы форматирования TeX и LaTeX.

Вы обязательно должны взять набор "A", остальные по желанию. Мы советуем инсталлировать наборы A, AP и D, а также X, если вы планируете использовать X Window System.

Получение Slackware из Internet Версии Slackware Linux на многих FTP-серверах мира. В Приложении C дан список некоторых из них. Мы предлагаем вам найти ближайший к вам FTP-сервер, чтобы минимизировать трафик. Тем не менее есть два главных Linux FTP-архива: `sunsite.unc.edu` и `tsx-11.mit.edu`.

Версию Slackware можно найти по крайней мере на следующих серверах:

- sunsite.unc.edu:/pub/Linux/distributions/slackware
- tsx-11.mit.edu:/pub/linux/packages/slackware
- ftp.cdrom.com:/pub/linux/slackware

`ftp.cdrom.com` - это родной сервер Slackware.

Перекачка файлов При перекачке файлов вам следует перекачать, используя FTP, следующие файлы. Не забывайте использовать режим перекачки двоичных файлов. Приложение C содержит достаточно материалов для того, чтобы научиться использовать FTP.

- Различные файлы `README`, а также `SLACKWARE_FAQ`. Прочитайте обязательно эти файлы до инсталляции.

- Образ bootdisk. Это файл, который вы запишите на дискету для создания загрузочного диска Slackware. Если у вас дисковод на 1.44M (3.5”), найдите в каталоге `bootdsks.144`. Если у вас дисковод 1.2M (5.25”), найдите в каталоге `bootdsks.12`.

  Вам необходим один из следующих файлов bootdisk.

  - bare.gz. Это загрузчик с дискет, имеющий только драйвер жесткого диска IDE. (Нет SCSI, CD-ROM или сетевой поддержки). Используйте, если у вас только есть контроллер IDE жесткого диска, а инсталляция по сети или с CD-ROM невозможна.
  - cdu31a.gz. Содержит драйверы IDE, SCSI и Sony CDU31A/33A.
  - mitsumi.gz. Содержит драйверы IDE, SCSI и Mitsumi CD-ROM
  - modern.gz. Экспериментальный загрузочный диск с новым ядром и всеми CD-ROM драйверами, кроме сетевых и Sony 535.
  - net.gz. Содержит CD-ROM драйвер IDE и сетевые.
  - sbpcd.gz. Содержит CD-ROM драйверы IDE, SCSI и SoundBlaster Pro/Panasonic.
  - scsi.gz. Содержит CD-ROM драйверы IDE, SCSI и SCSI
  - scsinet.gz. Содержит CD-ROM и сетевые драйверы IDE, SCSI, SCSI.
  - sony535.gz. Содержит CD-ROM драйверы IDE, SCSI и Sony 535/531
  - xt.gz. Содержит CD-ROM драйверы IDE и жесткого диска XT.

  Вам нужен *только один* образ диска из вышеперечисленных в зависимости от вашей аппаратуры.

  Некоторые драйверы конфликтуют друг с другом самым странным образом. И вместо того, чтобы выискивать ошибки в своей аппаратуре, проще использовать загрузочную дискету с конкретными драйверами. Большинству пользователей следует начинать попытки с `scsi.gz` или `bare.gz`.

- Образ root-диска. Это файл, который вы запишите на дискету для создания инсталляционного диска Slackware. В связи с этим образом посмотрите `rootdsks.144` или `rootdsks.12` в зависимости от типа дисковода, с которого производите загрузку.

  Вам необходим один из следующих файлов:

  - color144.gz. Инсталляционный диск для дисковода 1.44, использующий меню. Большинству пользователей следует использовать именно его.

- umsds144.gz. Версия диска `color144` для инсталляции с помощью файловой системы UMSDOS, которая позволяет инсталлировать Linux поверх каталога файловой системы MS-DOS. Этот метод инсталляции обсуждается здесь детально, но он не позволяет переразбивать диск на новые разделы. Позже мы к этому вернемся.
- tty144.gz. Инсталляционный диск, ориентированный на терминал, для дисковода 1.44M. Если color144.gz у вас не идет, попробуйте вместо него `tty144.gz`.
- colrlite.gz. Инсталляционный диск, ориентированный на терминал, для дисковода 1.2M.
- umsds12.gz. Версия диска `colrlite` для инсталляции с помощью файловой системы UMSDOS. Смотрите вышеприведенное описание `umsds144.gz`.
- tty12.gz Инсталляционный диск, ориентированный на терминал, для дисковода 1.2M. Используйте этот диск, если у вас есть дисковод на 1.2M и `colrlite.gz` у вас не идет.

И снова вам необходим *лишь один* образ диска root, зависящий от типа драйвера загрузочного дисковода.

- GZIP.EXE. Это MS-DOS программа, выполняющая компрессию `gzip`, используемая для сжатия файлов boot и root (расширение имен файлов ".gz" говорит об этом). Она может быть найдена в каталоге `install`.
- RAWRITE.EXE. Это MS-DOS программа, которая записывает содержимое файла (например, образов boot и root) прямо на дискету, не проверяя формат. Вам следует использовать `RAWRITE.EXE` для создания дискет boot и root. Это можно также найти в инсталляционном каталоге.

  Если вы планируете создать дискеты boot и root из MS-DOS, вам понадобятся только `RAWRITE.EXE` и `GZIP.EXE`. Если же вам доступна UNIX-станция с дисководом, вы можете создать дискеты в ней, используя команду `dd`. Посмотрите руководство по команде `dd` или посоветуйтесь со специалистами по UNIX.
- Файлы из каталогов `slakware`a1/,`slakware`a2/ и `slakware`a3/. Эти файлы создают набор диска "A" дистрибутива Slackware. Они необходимы. Позже вы скопируете эти файлы на дискеты в MS-DOS для инсталляции (или вы можете инсталлировать с жесткого диска). Поэтому, когда вы скачаете эти файлы, сохраните их в отдельном каталоге; не перепутайте файлы `a1` с файлами `a2` и т.п.

Убедитесь также, что вы получили файлы без точек в именах. В FTP используйте команду “`mget *`” вместо “`mget *.*`”.

- Файлы в каталогах `ap1`, `ap2` и т.д. зависят от инсталлируемых наборов. Например, если вы инсталлируете диск с набором “`X`”, берите файлы из каталогов от x1 до x5. Как и для наборов диска “`A`”, убедитесь, что файлы при переписывании размещаются в разных каталогах.

Slackware на CD-ROM Slackware можно также получить на CD-ROM. Большинство CD-ROM со Slackware просто содержат копии файлов, которые появляются в архивах FTP-серверов. Поэтому, если у вас есть CD-ROM со Slackware, значит у вас есть все необходимые файлы. Вы должны будете создать boot и root дискеты из файлов, взятых с CD-ROM. Смотрите Раздел 2.1.4.2.1.

Прежде всего следует решить, какие образы дисков boot и root будут использованы. Они должны быть на CD-ROM. Ниже мы опишем, как создавать эти дискеты.

Методы инсталляции Slackware поддерживает несколько вариантов инсталляции. Наиболее популярный - инсталляция из раздела MS-DOS вашего жесткого диска; другой способ инсталлировать с дискет в формате MS-DOS, созданных с дискового набора, который вы переписали.

Если у вас есть Slackware на CD-ROM, вы можете инсталлировать файлы прямо с него. Дистрибутив Slackware Pro от Morse Telecommunications позволяет инсталлировать Slackware так, что многие файлы доступны прямо с CD-ROM. Это может сэкономить много пространства на диске, но некоторые приложения будут работать медленнее.

Создание boot и root дискет Создавая дискеты boot и root, вы должны их создавать с образов boot-диска и root-диска, которые вы скачали (или имеете на CD-ROM), вне зависимости от вида инсталляции. В MS-DOS вы должны раскомпрессировать образы boot-диска и root-диска, используя GZIP.EXE. Например, если вы используете образ диска boot - `bare.gz`, наберите команду MS-DOS:

```
C: GZIP -D BARE.GZ
```

которая раскомпрессирует `bare.gz` и создаст вам файл `bare`. Если вы инсталлируете с CD-ROM, вы можете скопировать образ диска bootdisk (например, `bare.gz`) на жесткий диск и выполнить `GZIP.EXE` с CD-ROM для раскомпрессирования.

Вы должны также раскомпрессировать образ диска root. Например, если вы используете root-диск `color144.gz`, наберите команду:

```
C: GZIP -D COLOR144.GZ
```

которая раскомпрессирует этот файл и создаст файл `color144`.

Далее, вы должны иметь две high-density дискеты, отформатированные в MS-DOS. (Они должны быть одного типа; если ваша boot-дискета 3.5", обе дискеты должны быть high-density 3.5"). Для записи образов дисков boot и root на дискеты надо использовать `RAWRITE.EXE` .

Наберите команду:

```
C: RAWRITE
```

Ответьте на вопросы о имени переписываемого файла (например, `bare` или `color144`) и дисковода (например A:). `RAWRITE` скопирует файл блок за блоком прямо на дискету. Используйте `RAWRITE` также для образа root-диска. Когда вы это сделаете, у вас будут две дискеты: одна содержит boot-диск, другая root-диск. Имейте в виду, что эти две дискеты уже нечитаемы в MS-DOS (они уже, в известном смысле, в "Linux-формате").

Убедитесь, что вы используете хорошие несбойные дискеты. На дискетах не должно быть bad-блоков.

Обратите внимание на то, что вам не обязательно использовать MS-DOS для инсталляции Slackware. Но использование MS-DOS облегчает создание boot и root дискет, облегчает инсталляцию программ (поскольку вы можете инсталлировать прямо из MS-DOS раздела вашей системы). Если у вас на компьютере нет MS-DOS, вы можете использовать чужой компьютер для создания дискет, и уже инсталлировать с них.

Нет также необходимости использовать `GZIP.EXE` и `RAWRITE.EXE` под MS-DOS для создания дискет boot и root. Вы можете использовать команды gzip и dd в UNIX для выполнения той же работы. (Для этого, разумеется, вам нужна UNIX-станция с дисководом). Например, на станции Sun с дисководом `/dev/rfd0` вы можете использовать команды:

```
$ gunzip bare.gz
$ dd if=bare of=/dev/rfd0 obs=18k
```

Вы должны указать соответствующий размер блока (obs), иначе

на некоторых станциях (на тех же Sun) эта команда не будет выполнена. Если у вас возникнут проблемы - читайте руководство по команде `dd`.

Подготовка к инсталляции с жесткого диска. Если вы планируете инсталляцию Slackware прямо с жесткого диска (которая обычно и быстрее, и надежнее, чем с дискет), вам потребуется раздел MS-DOS.

Обратите внимание: Если вы планируете инсталляцию Slackware из раздела MS-DOS, этот раздел НЕ должен быть скомпрессирован с помощью DoubleSpace, Stacker или какой-то другой утилиты MS-DOS. В настоящее время Linux не может прямо читать MS-DOS-раздел DoubleSpace/Stacker. (Вы можете обратиться к ней через MS-DOS Emulator, но это не подходит при инсталляции Linux).

При подготовке инсталляции с жесткого диска создайте просто на жестком диске каталог для размещения файлов Slackware. Например,

```
C: MKDIR SLACKWAR
```

создаст каталог `C:`
`SLACKWAR` для хранения файлов Slackware. Под этим каталогом, используя команду `MKDIR`, вы можете создать подкаталоги `A1`, `A2` и т.д. для каждого переписанного дискового набора. Все файлы с диска `A1` должны быть помещены в каталог `SLACKWAR`
`A1` и т.д.

Подготовка к инсталляции с дискет. Если вы хотите инсталлировать Slackware с дискет, вместо жесткого диска, вам нужно по одной чистой дискете, отформатированной в MS-DOS, для каждого диска Slackware, который вы желаете переписать. Дискеты должны быть формата high-density.

Набор диска `A` (диски `A1` - `A3`) могут быть дискетами как 3.5", так и 5.25". Но остальные наборы дисков должны быть на дискетах 3.5".

Поэтому, если у вас есть только дисковод 5.25", необходимо взять у кого-нибудь на прокат дисковод 3.5", чтобы инсталлировать прочие (кроме A) диски. Или вы можете инсталлировать их с жесткого диска, как это описывалось выше.

Для того, чтобы создать диски, просто скопируйте файлы из каждого Slackware каталога на отформатированные в MS-DOS дискеты, используя команду MS-DOS - COPY:

```
C: COPY A1*.* A:
```

Которая скопирует содержимое диска `A1` на дискету в дисководе `A:`. Это следует повторить для всех считываемых дисков.

Нет необходимости каким-либо образом модифицировать или раскомпрессировать файлы диска; вы просто должны скопировать их на дискеты в MS-DOS. Процедура инсталляции Slackware сама заботится о раскомпрессировании файлов.

Подготовка к инсталляции с CD-ROM. Если у вас Slackware на CD-ROM, вы можете инсталлировать систему, как только вы создали дискеты boot и root. Программы будут инсталлироваться прямо с CD.

2.2 Подготовка к инсталляции Linux

После того, как вы получили дистрибутив Linux, вы можете готовить свою систему к инсталляции. Требуется спланировать работу, особенно если вы уже работали на других операционнных системах. В последующих разделах мы расскажем, как планировать инсталляцию Linux.

2.2.1 Общие принципы инсталляции

Хотя версии Linux отличаются, общие методы инсталляции состоят в следующем:

1. **(Пере)разбейте на разделы жесткий диск(и).** Если у вас уже инсталлирована другая операционная система, вы должны сделать *переразбиение*, чтобы выделить место под Linux. Это обсуждается в Разделе 2.2.4.
2. **Загрузите средства инсталляции Linux.** Каждый дистрибутив имеет в каком-либо виде средства инсталляции - обычно загрузочную (boot) дискету, которая используется для инсталляции программ. Загрузка этих средств либо представит вам некую пошаговую программу инсталляции, либо позволит инсталлировать вручную.
3. **Создайте разделы для Linux.** После переразбиения и выделения места под Linux, вы создаете на этом месте раздел Linux. Это выполняется программой Linux `fdisk`. (см. Раздел 2.3.3).
4. **Создайте файловые системы и область своппинга.** Вы создадите одну или несколько файловых систем для хранения файлов на вновь созданном разделе. Кроме того, если вы желаете получить область своппинга, то также создадите и его на одном из разделов Linux. (См. Разделы 2.3.4 и 2.3.5).
5. **Инсталлируйте программы Linux в новую(вые) файловую(вые) систему(мы).** Далее вас ждет спокойное плавание, если все

прошло нормально. (См. Раздел 2.3.6). Позже, в Разделе 2.5, мы опишем, что делать, когда что-то не получается.

Многие дистрибутивы Linux снабжаются инсталляционной программой, которая будет руководить вами в процессе инсталляции и автоматизирует некоторые из описанных шагов. Имейте в виду, что в различных дистрибутивах некоторые шаги могут быть автоматизированы.

Дистрибутивы Slackware для Linux, описываемые в этой книге, потребуют от вас лишь разбиения диска на разделы с использованием fdisk, а также использования setup для выполнения других шагов.

Важное замечание. При подготовке к инсталляции Linux, лучший совет, который мы можем дать - это делать записи в ходе инсталляции. Записывайте все, что вы вводите с клавиатуры и все, что видите неординарное. Смысл здесь простой: если (или когда!) вы попадете в тупик, вам будет важно проследить в обратном порядке ваши шаги, чтобы обнаружить ошибку. Инсталляция Linux несложна, но надо помнить множество деталей. Вам хорошо бы зафиксировать все эти детали, чтобы вы могли экспериментировать с другими методами, если что-то не будет получаться. Эти записи будут также полезны, если вы обратитесь к другим людям за помощью. Например, напишите в Linux-конференцию USENET. Эти заметки вам однажды захочется показать своим внукам.[9]

2.2.2 Концепция разбиения на разделы

В общем случае жесткие диски разбиваются на разделы, где отдельные разделы выделяются отдельным операционным системам. Например, вы можете сделать на диске несколько независимых разделов: один, скажем, для MS-DOS, другой для OS/2 и третий для Linux.

Если у вас уже есть инсталлированные программы, вам может потребоваться переразбиение диска, чтобы выделить место для Linux. Затем вы создадите на освободившемся месте один или несколько разделов для Linux и области свопинга.

Многие системы MS-DOS используют один раздел, заполняя все дисковое пространство. В MS-DOS этот раздел известен, как C:. Если у вас более одного раздела, MS-DOS дает им имена D:, E: и т.д. Каждый раздел выступает как независимый диск.

На первом секторе диска находится **master boot record** с таблицей разделов. **boot**

---

[9] Автор со смущением признает, что он хранит тетрадку с записью всех своих кувырканий с Linux в первые месяцы работы с системой. Сейчас эта тетрадка притягивает пыль на книжной полке.

record (загрузочная запись) используется для загрузки системы. Таблица разделов содержит информацию о местоположении и размере разделов.

Существует три типа разделов: первичные, расширенные и логические (**primary**, **extended** и **logical**). Наиболее часто используются первичные разделы. Однако, из-за ограничений на размер таблицы разделов, можно иметь не более четырех разделов на любом диске.

Чтобы обойти ограничение четырех разделов, используются расширенный раздел. Расширенные разделы сами по себе не содержат данных. Они выступают как хранилища логических разделов. Поэтому вы можете создать один расширенный раздел, покрывающий весь диск, и внутри создать много логических разделов.

2.2.3 Требования Linux к разделам

Прежде, чем мы расскажем, как переразбивать ваш диск, вы должны представлять, сколько места надо выделить под Linux. Как создавать эти разделы, мы обсудим позже в Разделе 2.3.3.

В системах UNIX файлы хранятся в **файловой системе**, которая прежде всего расположена на диске (или на другом устройстве, вроде CD-ROM или дискеты), отформатированном для хранения файлов. Каждая файловая система ассоциируется с конкретной частью дерева каталогов; например, во многих случаях существует файловая система для всех файлов каталога `/usr`, другая для `/tmp` и т.д. Корневая файловая система - первичная файловая система, которой соответствует самый верхний каталог `/`.

Под Linux каждая файловая система "живет" в отдельном разделе диска. Например, если у вас есть файловая система для `/` и другая для `/usr`, вам потребуется два раздела.

Прежде, чем инсталлировать Linux, вам необходимо подготовить файловые системы для размещения программ Linux. Вы должны иметь по крайней мере одну файловую систему (корневую файловую систему), а поэтому один раздел, назначенный для Linux. Многие пользователи Linux умудряются поместить все свои файлы в корневую файловую систему, с которой (одной) управляться легче, чем с множеством файловых систем.

Но вы можете создать много файловых систем для Linux, если пожелаете. Например, если вы хотите использовать раздельные файловые системы для `/usr` и `/home`. Читатели, имеющие опыт работы в качестве администраторов UNIX, знают, как извлекать пользу из раздельных файловых систем. В Главе 4 мы обсуждим, как разумно использовать множество файловых систем и разделов.

Зачем использовать больше, чем одну файловую систему? Наиболее очевидная причина - безопасность. Если по какой-то причине одна из ваших файловых систем будет повреждена, то другие, как правило, не пострадают. С другой стороны, если вы поместите все свои файлы в корневую файловую систему, и по какой-то причине она будет испорчена - все разом пропадет. Но это редкие случаи. Если вы регулярно делаете системный backup, то можете чувствовать себя достаточно безопасно.[10]

Другая причина использования множества файловых систем - это использование памяти нескольких дисков. Если у вас есть, скажем, 40M свободного места на одном диске и 50M на другом, вы можете создать корневую файловую систему на одном диске и `/usr` на 50-ти мегабайтном. В настоящее время невозможно одной файловой системе располагаться сразу на нескольких дисках, поэтому необходимо создавать несколько файловых систем.

Таким образом, можно сказать, что Linux требует не менее одного раздела для корневой файловой системы. Если вы желаете создать несколько файловых систем, вы должны для каждой выделить свой раздел. Некоторые дистрибутивы автоматически создают вам разделы и файловые системы, так что вы можете вообще об этом не беспокоиться.

Следует также иметь в виду проблемы своппинга. Если вы хотите использовать область своппинга в Linux, у вас две возможности. Во-первых, использовать *файл своппинга*, который существует в одной из файловых систем. Вы создадите файл своппинга для использования как виртуальной RAM после инсталляции. Во-вторых, создать *раздел своппинга*, который будет использоваться только для этого. Большинство использует второй способ.

Отдельный файл своппинга или раздел могут быть не более 16M. Если вы желаете использовать для своппинга более 16M, вы можете создать несколько разделов или файлов своппинга (до восьми). Например, если вам нужна область своппинга в 32M, вы можете создать два раздела по 16M.

Создание раздела своппинга обсуждается в Разделе 2.3.4, а создание файла своппинга - в Главе 4.

В общем случае вы создадите для Linux два раздела: один для корневой файловой системы и другой для области своппинга. Разумеется, возможно много вариаций на эту тему, но это минимальная конфигурация. Вы не обязаны иметь область своппинга для Linux, но если у вас менее 16M RAM, то это очень настоятельно рекомендуется.

Разумеется, вы должны представлять, сколько места потребуется выделить под этот раздел. Размер вашей файловой системы в большой степени зависит от используемого

---

[10] Автор использует одну (200M) файловую систему для всех своих файлов и не имеет проблем (пока).

дистрибутива и об'емов инсталляции. Надеемся, что сопутствующая дистрибутиву документация даст вам приблизительные оценки потребности в памяти. Маленькие Linux-системы могут занимать 20M и даже меньше. Большие системы - от 80M до 100M, и даже больше. Но не забывайте, что необходимо место также под каталоги пользователей и под будущие расширения.

Размер вашей области свопинга (если вам доведется выбирать) зависит от того, какая вам требуется виртуальная память. Хорошее прикидочное правило: область свопинга равняется удвоенной RAM. Например, у вас 4M RAM (физической памяти), для нее хорошо иметь раздел в 8M для области свопинга. Но это несколько абстрактные рассуждения, поскольку область свопинга зависит в значительной степени от эксплуатируемых программ. Если у вас много физической памяти (скажем, 16M и даже более), может быть вам вообще не захочется использовать область свопинга.

Важное замечание. Из-за ограничений BIOS часто невозможно загрузиться из разделов, использующих номера цилиндров выше 1023. Поэтому, выделяя место для Linux, имейте в виду, что вы можете не захотеть использовать разделы с номерами цилиндров больше 1023 для корневой файловой системы. Но Linux *может* эти разделы использовать (но не для *загрузки*). Может быть этот совет и преждевременный, но это важно для планирования диска.

Если вы вынуждены использовать разделы с номерами цилиндров больше 1023 для корневой файловой системы, вы всегда можете загружать Linux с дискеты. Это не так плохо, действительно, это дольше всего на несколько секунд по сравнению с загрузкой с жесткого диска. В любом случае это всегда запасной вариант.

2.2.4 Разбиение вашего жесткого диска

В этом разделе мы опишем, как изменить размеры ваших уже существующих разделов, чтобы выделить место для Linux. Если вы инсталлируете Linux на "чистый" диск, вы можете пропустить этот раздел и продолжить чтение с Раздела 2.3.

Обычный способ изменить размеры существующего раздела - это удалить его (уничтожив все данные, которые он содержал) и вновь его создать. Перед переразбиением диска *сделайте backup системы*. А затем восстановите информацию. В MS-DOS существует несколько программ, которые могут переразбить диск без уничтожения информации. Одна из них известна как `FIPS`, ее можно найти на FTP-серверах Linux.

Имейте также в виду, что, поскольку вы будете уменьшать существовавшие разделы, у вас может потом не хватить места, чтобы все восстановить. В этом случае вы предварительно должны стереть все маловажное, чтобы после уменьшения разделов хватило места.

Программа, используемая для разбиения на разделы известна, как `fdisk`. Каждая операционная система имеет собственный аналог этой программы; например, под MS-DOS она вызывается командой `FDISK`. Вам следует посмотреть документацию на вашу операционную систему относительно перераспределения разделов. Здесь мы обсудим, как переразбивать на разделы в MS-DOS с использованием `FDISK`, но эта информация может быть легко распространена на другие операционные системы.

Посмотрите документацию на свою операционную систему, прежде чем переразбивать свой диск. Этот раздел затрагивает лишь основные черты переразбиения; существует много тонкостей, которые здесь не рассматриваются. Вы можете потерять все программы, если неправильно переразобьете свой диск.

Предупреждение. Не модифицируйте и не создавайте вновь раздел для любой другой операционной системы (включая Linux), используя `FDISK` под MS-DOS. Вы можете производить модификацию только для конкретной операционной системы; например, вы должны создавать разделы для Linux, используя версию `fdisk` для Linux. Позже, в Разделе 2.3.3 мы опишем, как создавать разделы для Linux. Но сейчас мы обсуждаем изменение размера существующих.

Предположим, что вы имеете один диск, полностью отданный под MS-DOS. Таким образом, ваш диск состоит целиком из одного раздела, обычно известного как “`C:`”. Поскольку этот метод портит данные этого раздела, вам необходимо создать загрузочную таблицу MS-DOS “system disk”, которая хранит все необходимое для выполнения `FDISK` и последующего backup.

Во многих случаях вы можете использовать для этой цели инсталляционные диски под MS-DOS. Если вы хотите создать свой системный диск, отформатируйте дискету командой

```
FORMAT /s A:
```

Скопируйте на эту дискету все необходимые утилиты MS-DOS (обычно большинство программ каталога
DOS на вашем диске), а также программы `FORMAT.COM` и `FDISK.EXE`. Теперь вы можете загрузиться с этой дискеты и выполнить команду

```
FDISK C:
```

Работа `FDISK` должна сопровождаться об'яснениями, но детали следует посмотреть в документации на MS-DOS. Когда вы запустите `FDISK`, используйте опцию menu для отображения таблицы разделов и спишите приведенную там информацию. Важно сохранить запись о ваших первоначальных установках setup в случае, если вы захотите отказаться от инсталляции Linux.

Для удаления существующего раздела выберите в меню FDISK опцию "`Delete an MS-DOS Partition or Logical DOS Drive`". Опишите тип раздела, который вы желаете удалить (первичный, расширенный или логический) и число разделов. Внимательно отнеситесь ко всем предупреждениям. Ух!

Для создания нового (меньшего) раздела для MS-DOS, выберите опцию FDISK "`Create an MS-DOS Partition or Logical DOS Drive`". Опишите тип раздела (первичный, расширенный или логический) и размер создаваемого раздела (в мегабайтах). `FDISK` создаст раздел.

После того, как вы закончите работу с `FDISK`, следует выйти из программы и переформатировать новый раздел. Например, если вы изменяете размер первого DOS раздела на диске (`C:`), следует выполнить команду

```
FORMAT /s C:
```

Теперь можно выполнить backup.

2.3 Инсталляция Linux

После того, как вы переразбили диск, чтобы выделить место под Linux, вы можете начать инсталляцию. Здесь дано краткое описание процедур:

- Загрузите средства инсталляции Linux;
- Выполните fdisk под Linux для создания разделов;
- Выполните mke2fs и mkswap для создания файловой системы Linux и области своппинга;
- Инсталлируйте программы Linux;
- Наконец, инсталлируйте либо загрузчик boot-файла LILO на диске, или создайте загрузочную дискету для загрузки новой Linux-системы.

Как мы уже сказали, некоторые шаги могут быть автоматизированы, в зависимости от используемого дистрибутива Linux. Пожалуйста, посмотрите более подробные инструкции в документации.

2.3.1 Загрузка Linux

Первый шаг - загрузить средства инсталляции Linux. В большинстве случаев это **загрузочная дискета**, которая содержит маленькую Linux-систему. При загрузке с дискеты вам будет пред'явлено в каком-то виде меню, которое поможет вам в процессе инсталляции. В других дистрибутивах при загрузке дискеты выдается подсказка login. В этом случае вы обычно входите как root и начинаете процесс инсталляции.

В документации, сопровождающей дистрибутив говорится, что необходимо делать для загрузки Linux.

Если вы инсталлируете дистрибутив Slackware, то все, что требуется, это загрузить загрузочную дискету, которую вы создали, следуя предыдущему разделу.

Большинство дистрибутивов Linux используют загрузочную дискету, которая позволяет ввести параметры компьютера при загрузке, для определения особенностей устройств. Например, если ваш SCSI-контроллер не распознается при загрузке дискеты, вы должны перезагрузиться и описать параметры аппаратуры (например, I/O-адрес и IRQ).

Похоже, что компьютеры IBM PS/1, ThinkPad и ValuePoint не сохраняют геометрию диска в CMOS и вы должны ее описывать во время загрузки.

Подсказка boot часто выдается автоматически, когда загружается загрузочная дискета. Так, например, происходит при загрузке в дистрибутиве Slackware. Некоторые дистрибутивы потребуют от вас удерживать во время загрузки с дискеты *shift* или *ctrl*. В случае успеха вы должны увидеть подсказку:

```
boot:
```

и, возможно, другие сообщения.

Можно попробовать загрузиться без каких-либо специальных параметров, просто нажать *enter* в ответ на подсказку boot.

Следите за сообщениями во время загрузки. Если у вас SCSI контроллер, вы увидите список распознаваемых устройств (hosts). Если вы увидите сообщение

```
SCSI: 0 hosts
```

это значит, что ваш SCSI контроллер не был опознан, и вам следует использовать другую процедуру.

Система также представит информацию о разделах диска и распознанных устройствах. Если какая-либо информация неверна или отсутствует, вы должны инициировать распознавание оборудования.

С другой стороны, если все идет хорошо и оборудование, вроде бы, распознается, вы можете перейти к следующему разделу, к Разделу 2.3.2.

Для инициации распознавания оборудования вы должны ввести соответсвующие параметры после подсказки загрузчика, используя следующий синтаксис:

```
ramdisk parameters...
```

Существует ряд доступных параметров: вот некоторые наиболее характерные.

hd=<cylinders>,<heads>,<sectors>

Описывает геометрию для таких систем, как IBM PS/1, ValuePoint и ThinkPad. Например, если у вашего диска 683 цилиндров, 16 головок и 32 сектора на трек, введите per track, enter

ramdisk hd=683,16,32

tmc8xx=<memaddr>,<irq>

Описывает адрес и IRQ для без-BIOS-ных Future Domain TMC-8xx SCSI контроллеров. Например,

```
ramdisk tmc8xx=0xca000,5
```

Обратите внимание, что префикс "0x" должен использоваться для всех значений, данных в шестнадцатиричной системе. Это справедливо для всех последующих опций.

st0x=<memaddr,<irq>

Описывает адрес и IRQ для без-BIOS-ных Seagate ST02 контроллеров.

t128=<memaddr>,<irq>

Описывает адрес и IRQ для без-BIOS-ных Trantor T128B контроллеров.

ncr5380=<port>,<irq>,<dma> Описывает порт, IRQ и DMA канал для generic NCR5380 контроллера.

aha152x=<port>,<irq>,<scsi_id>,1

Описывает порт, IRQ и SCSI ID для без-BIOS-ных AIC-6260 контроллеров. Включает Adaptec 1510, 152x и Soundblaster-SCSI контроллеры.

Для каждого из них вы должны ввести ramdisk с параметрами, которые вы хотите установить.

Если у вас есть вопросы относительно опций периода загрузки, посмотрите Linux *SCSI HOWTO*, который можно найти на любом Linux FTP-сервере (или там, где вы раздобыли эту книгу), а также Linux *CD-ROM HOWTO*. Эти документы описывают возможности аппаратуры более детально.

2.3.2 Дисководы и разделы под Linux

Многие дистрибутивы предполагают ручное создание разделов Linux с использованием программы `fdisk`. Другие могут автоматически создавать разделы. В любом случае вы должны знать о существовании разделов и имен дисководов. Дисководы и разделы под Linux имеют другие имена, по сравнению с другими операционными системами. Под MS-DOS дисководы гибких дисков именуются `A:` и `B:`, в то время, как разделы жесткого диска именуются `C:`, `D:`, и т.д. В Linux соглашение о именах совсем другое.

Драйверы устройств,находящиеся в каталоге `/dev`, используются для общения с устройствами системы (такими, как жесткий диск, мышь и т.п.)

Например, если у вас есть мышь, вы имеете к ней доступ через драйвер `/dev/mouse`. Дисководы гибких дисков, жестких дисков и отдельные разделы имеют индивидуальные драйверы. Пока можете не беспокоиться относительно интерфейса; пока важно только понять, как именуются различные устройства при их использовании.

Таблица 2.1 содержит имена различных драйверов.

Несколько замечаний по поводу этой таблицы. Обратите внимание, что `/dev/fd0` соответствует первому дисководу (`A:` для MS-DOS) и `/dev/fd1` соответствует второму дисководу (`B:`).

SCSI-драйверы жестких дисков названы иначе, чем другие драйверы. Драйверы IDE, MFM и RLL доступны через устройства `/dev/hda`, `/dev/hdb` и т.д. Индивидуальные разделы на дисководе `/dev/hda` будут `/dev/hda1`, `/dev/hda2` и т.д. А SCSI-драйверы именуются `/dev/sda`, `/dev/sdb`, и т.д., с разделами, именуемыми `/dev/sda1` и `/dev/sda2`.

Вот например. Предположим, что у вас один IDE-дисковод с тремя первичными разделами. Два первых выделены под MS-DOS, а третий раздел - расширенный, содержащий два логических раздела под Linux. Устройства, соответствующие этим разделам, будут:

```
Первый раздел MS-DOS (C:)     /dev/hda1
```

Второй раздел MS-DOS (D:)	/dev/hda2
Расширенный раздел	/dev/hda3
Первый логический раздел Linux	/dev/hda5
Второй логический раздел Linux	/dev/hda6

Обратите внимание, что пропущен `/dev/hda4`, он соответствует четвертому первичному разделу, которого нет в этом примере. Логические разделы именуются, начиная с `/dev/hda5`.

Устройство	Имя
Первый дисковод (A:)	/dev/fd0
Второй дисковод (B:)	/dev/fd1
Первый жесткий диск (весь)	/dev/hda
Первый жесткий диск, первичный раздел 1	/dev/hda1
Первый жесткий диск, первичный раздел 2	/dev/hda2
Первый жесткий диск, первичный раздел 3	/dev/hda3
Первый жесткий диск, первичный раздел 4	/dev/hda4
Первый жесткий диск, логический раздел 1	/dev/hda5
Первый жесткий диск, логический раздел 2	/dev/hda6
..	
Второй жесткий диск (весь)	/dev/hdb
Второй жесткий диск, первичный раздел 1	/dev/hdb1
..	
Первый SCSI драйвер диска (весь)	/dev/sda
Первый SCSI драйвер диска, первичный раздел 1	/dev/sda1
..	
Второй SCSI драйвер диска (весь)	/dev/sdb
Второй SCSI драйвер диска, первичный раздел 1	/dev/sdb1
..	

Табл. 2.1: Имена разделов Linux

2.3.3 Создание разделов Linux

Теперь вы готовы создать разделы Linux с помощью команды `fdisk`. Как описывалось в Разделе 2.2.3, в общем случае вам необходимо создать как минимум один раздел для самого Linux и другой для области свопинга.

После загрузки средств инсталляции выполните команду `fdisk`, напечатав

```
fdisk drive
```

где <drive> имя устройства в Linux, которому вы хотите выделить раздел (см. Табл.

2.1). Например, если вы хотите выполнить fdisk для первого SCSI-диска, используйте команду `fdisk /dev/sda`.

`/dev/hda` (первый IDE-диск) берется по умолчанию, если вы не описали другого.

Если вы создаете разделы для Linux более, чем на одном диске, выполните fdisk отдельно для каждого диска.

```
# fdisk /dev/hda

Command (m for help):
```

В этот момент fdisk ждет команды; вы можете ввести "m", чтобы получить перечень опций.

```
Command (m for help):  m
Command action
a toggle a bootable flag
d delete a partition
l 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):
```

Для создания нового раздела используется команда `n`. О большинстве других опций вы можете не вспоминать. Выйти из программы `fdisk`, без сохранения произведенных изменений, можно командой `q`. Выйти из программы `fdisk` с записью изменений в таблице разделов можно командой `w`.

Первое, что вы должны сделать, это получить и записать текущее состояние таблицы разделов. Используйте команду `p`.

```
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):
```

Это пример, когда у нас один MS-DOS-раздел на `/dev/hda1`, который имеет 61693 блоков (около 60M - блок в Linux - 1024bytes). Этот раздел начинается на цилиндре N 1 и заканчивается на цилиндре N 203. Всего у нас на диске 683 цилиндров. Так что остается 480 цилиндров для создания раздела Linux.

Для создания нового раздела используйте команду "n". В этом примере мы создадим два новых первичных раздела (`/dev/hda2` and `/dev/hda3`) для Linux.

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

Здесь `fdisk` спрашивает тип создаваемого раздела: extended (расширенный) или primary (первичный). В нашем примере мы создаем только первичный раздел, так что выбираем `p`.

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

Затем `fdisk` спросит число создаваемых разделов; поскольку раздел 1 уже использован, наш первый раздел Linux получит номер 2.

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

Теперь введите номер первого цилиндра раздела. Поскольку цилиндры с 204 по 683 не используются, мы используем первый свободный (номер 204). Нет смысла оставлять пустые места между разделами.

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

Программа `fdisk` запрашивает размер создаваемого раздела. Мы можем указать последний номер свободных цилиндров или размер в байтах, килобайтах или мегабайтах. Поскольку мы хотим, чтобы наш раздел был размером в 80M, мы укажем +80M. При указании размера раздела таким способом `fdisk` округлит действительный размер раздела до ближайшего числа цилиндров.

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

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

Если вы увидите предупреждение, вроде этого, его можно проигнорировать. Программа **fdisk** выдает сообщение, поскольку это старая программа, написанная еще до того, когда в Linux были разрешены разделы более, чем 64M. Теперь мы готовы создать второй раздел для Linux. С целью демонстрации мы создадим его размером в 10M.

```
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
```

Наконец, мы выдадим таблицу разделов. Вновь запишите всю информацию, особенно размеры в блоках ваших новых разделов. Вам потребуется знать размер разделов позже при создании файловой системы. Попутно проверьте, что разделы не накладываются друг на друга.

```
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   81  Linux/MINIX
/dev/hda3      474    474   507  10336   81  Linux/MINIX
```

Как вы видите, теперь есть **/dev/hda2** - раздел размером в 82080 блоков (что соответствует приблизительно 80M), и **/dev/hda3** - 10336 блоков (около 10M). прим.переводчика : видимо, уавтораздесьслова”сектор”и”блок” — синонимы

Имейте в виду, что много дистрибутивов (вроде того же Slackware) требуют использования команды **t** в программе **fdisk** для изменения области своппинга “Linux swap”, которая обычно имеет номер 82. Вы можете воспользоваться командой **L** для печати кодов типов известных разделов, а затем использовать **t**, чтобы установить тип области своппинга, соответствующий “Linux swap”.

При этом программы инсталляции смогут автоматически найти ваши разделы своппинга, основываясь на типе. Если ваши программы инсталляции не распознают области своппинга, вы можете снова запустить программу `fdisk` и использвать команду "t" в режиме вопросов.

В вышеприведенном примере оставшиеся цилиндры диска (номера с 508 по 683) не использованы. Вы можете позже создать дополнительные разделы.

Наконец, мы используем команду `w`, чтобы записать изменения и выйти из `fdisk`

```
Command (m for help):  w

#
```

Имейте в виду, что ни одно из сделанных изменений не даст эффекта то тех пор, пока вы не дадите команду `w`. Так что вы можете играть с различными конфигурациями и сохранить их, когда закончите. Кроме того, если вы захотите выйти из `fdisk` в любое время без сохранения изменений, используйте команду `q`. Помните, что программой `fdisk` для Linux вы можете выделять разделы для Linux и **только** для Linux.

Не забывайте также, что вы не сможете загружать Linux с разделов, использующих номера цилиндров, превышающие 1023. Поэтому, вы должны попытаться создать корневой раздел Linux на цилиндрах до 1023-го. Но если это невозможно - загружайтесь с дискеты.

Некоторые дистрибутивы Linux требуют перезагрузки системы после окончания работы `fdisk`. Это позволяет изменениям в таблице разделов оказать свое влияние на последующую инсталляцию. Новые версии `fdisk` автоматически изменяют сответствующую информацию в ядре, так что перезагрузка не требуется. Чтобы обезопасить себя, после выполнения `fdisk` вам следует снова загрузить средства инсталляции как и раньше - перед продолжением инсталляции.

2.3.4 Создание области своппинга

Если вы планируете использовать раздел своппинга для виртуальной памяти, вы должны быть готовы к его использованию.[11] В Главе 4 мы обсудим подготовку файла своппинга в случае, если вы не захотите использовать отдельный раздел.

Многие дистрибутивы потребуют от вас создать и активизировать область своппинга до инсталляции программ. Если у вас небольшой об'ем физической RAM, процесс

---

[11] (Некоторые дистрибутивы подготавливают область своппинга автоматически или через соответствующую опцию меню инсталляции).

инсталляции может не завершиться успешно, пока вы не выделите какой-то об'ем под область своппинга.

Дистрибутив Slackware требует создания области своппинга до инсталляции, если вы имеете 4M RAM или меньше. Если у вас нет таких ограничений, инсталляционная процедура Slackware выделит область своппинга автоматически. Если у вас возникают сомнения, то следуйте процедуре, описанной здесь; она не сможет вам навредить.

Команда создания раздела для своппинга называется `mkswap` и имеет вид

```
mkswap -c partition size
```

где <partition> - имя раздела своппинга, а <size> - размер этого раздела в блоках. size of the partition, in blocks.[12]

Например, если ваш раздел своппинга `/dev/hda3` и имеет размер в 10336 блоков, используйте команду

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

Опция `-c` команды `mkswap` обеспечивает проверку плохих блоков в разделе при создании области своппинга.

Если вы используете несколько разделов для своппинга, вам необходимо выполнить соответствующие команды `mkswap` для каждого раздела.

После форматирования области своппинга необходимо сделать ее пригодной для использования системой. Обычно система автоматически готовит место во время загрузки. Но, поскольку вы еще не инсталлировали Linux, вы должны подготовить его вручную.

Команда подготовки области своппинга - `swapon` имеет вид

```
swapon partition
```

Для вышеприведенного примера, чтобы подготовить область своппинга на `/dev/hda3`, необходимо выполнить команду

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

[12] Еще раз напомним, что в некоторых дистрибутивах область своппинга создается автоматически и блок в Linux занимает 1024 байта.

2.3.5 Создание файловых систем

Перед тем, как вы сможете использовать разделы Linux для хранения файлов, вы должны создать на них **файловые системы**. Создание файловой системы аналогично форматированию раздела под MS-DOS. Мы кратко обсуждали файловые системы в Разделе 2.2.3.

В Linux возможно несколько типов файловых систем. Каждый тип файловой системы имеет свой формат и характеристики (такие как имя файла, длина, максимальный размер файла и т.д.). Linux также поддерживает "третьи" типы файловых систем, например файловую систему MS-DOS.

Наиболее популярный тип файловой системы - это **Second Extended Filesystem** или *ext2fs*. *ext2fs* - одна из наиболее эффективных и гибких файловых систем. Она допускает использование имен файлов до 256 символов и размер файловой системы до 4 терабайтов (прим. переводчика: знать-то очень много). В Главе 4 мы обсудим различные типы файловых систем, возможных в Linux. А первоначально мы предполагаем, что вы используете файловую систему *ext2fs*.

При инсталляции дистрибутива Slackware файловые системы создаются автоматически инсталляционной процедурой, описываемой в следующем разделе. Если вы хотите создать файловую систему вручную, следуйте процедуре, описанной здесь.

Для создания файловой системы используйте команду

```
mke2fs -c partition size
```

где <partition> - имя раздела, а <size> - его размер в блоках. Например, для создания файловой системы из 82080 блоков на `/dev/hda2`, используйте команду

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

Если вы используете различные файловые системы для Linux, вам надо использовать соответствующую команду mke2fs для каждой файловой системы.

Если вы столкнулись с проблемами, смотрите Раздел 2.5.

2.3.6 Инсталляция программ

Наконец вы готовы инсталлировать программы. Каждая дистрибуция имеет для этого свой механизм. Многие дистрибутивы имеют самодокументированные программы, которые помогают пользователю пройти весь путь инсталляции. В других дистрибутивах вы должны **примонтировать** файловую систему к конкретному каталогу (например, `/mnt`) и вручную скопировать туда программы. В дистрибутивах на CD-ROM вам может быть дана опция инсталлирования части программ на жестком диске, оставив большую часть на CD-ROM.

Некоторые дистрибутивы предлагают несколько различных путей инсталляции. Например, вам может быть предоставлена возможность инсталлировать прямо из раздела MS-DOS вашего жесткого диска вместо дискет. Или вы будете иметь возможность инсталлировать по TCP/IP через FTP или NFS. Детали смотрите в документации на ваш дистрибутив.

Например, дистрибутив Slackware требует от вас только создания разделов с помощью `fdisk`, возможно, создания области своппинга с помощью `mkswap` и `swapon` (если у вас RAM 4M и меньше), а затем запуска программы `setup`. `setup` представит вам меню с об'яснениями. Использование setup в деталях описано ниже.

Конкретный метод инсталляции может существенно отличаться в различных дистрибутивах. Мы надеемся, что инсталляция Linux будет происходить с достаточными комментариями системы (что имеет место в большинстве дистрибутивов).

Инсталляция Slackware с `setup` Если вы инсталлируете Slackware, после создания разделов (возможно, и области своппинга) используйте команду

```
# setup
```

Она предоставит вам меню, с помощью которого будет направлять вас в процессе инсталляции.

Процедура, описанная здесь, соответствует находящейся на корневых дисках `color144` и `colrlite`; другие корневые диски могут иметь несколько отличающиеся процедуры.

Меню `setup` состоит из следующих пунктов. Используйте стрелки для перемещения по пунктам и нажимайте *enter* или *spacebar* при выборе пункта.

Help

Выводит help-файл программы `setup`.

Keymap

Эта опция позволяет описывать отображение клавиатуры в вашей системе, если у вас не US-клавиатура. Список keymaps (образов клавиатур) будет представлен; выберите соответствующий пункт из списка.

Quick

Позволяет выбирать между режимами “quick” (быстро) и “verbose”(с подробными комментариями). “Verbose” - режим, устанавливаемый по умолчанию, рекомендуется, если только вы не занимались уже этим десятки раз.

Make tags

Позволяет экспертам по инсталляции Slackware создавать настроенные tag-файлы. Это необходимо только для настройки инсталляционной процедуры; но это все не ваша забота.

Addswap

Это будет первый пункт, который выберет большинство пользователей при инсталляции Slackware. Будет пред'явлен список доступных разделов для своппинга (эти разделы будут типа “Linux swap” как наборы в `fdisk`). Вы сможете описать, какие разделы вы хотите использовать для области своппинга. Затем вам будет задан вопрос, хотите ли использовать mkswap для этих разделов.

Если вы уже выполнили `mkswap` и `swapon` (как описано в Разделе 2.3.4) над своими разделами своппинга, то нельзя в `setup` выполнять над этими разделами mkswap.

Даже если вы уже выполнили `mkswap` и `swapon`, необходимо использовать пункт Addswap меню: это гарантирует, что разделы своппинга будут доступны по завершении инсталляции.

! **Предупреждение!** Создание области своппинга в разделе разрушает данные, находившиеся на этом месте. Убедитесь, что вы не уничтожаете данные, которые следует сохранить. Если вы выбрали этот пункт меню, вам автоматически будут выдаваться подсказки. В общем случае это целесообразно.

Target

Этот пункт позволяет описывать разделы, на которые будет инсталлироваться Linux. Список доступных разделов (с типом “Linux native” (”исходный Linux” - описывается командой `fdisk`) будет отображен и вас попросят ввести имя корневого раздела Linux, например `/dev/hda2`. Далее вы получите подсказку относительно типа создаваемой файловой системы. Мы предлагаем использовать файловую систему типа `ext2fs`, как описано в Разделе 2.3.5. Это создаст файловую систему в названном разделе, нечто аналогичное форматированию раздела под MS-DOS. Вы также получите подсказки относительно любых других разделов, которые вы можете захотеть использовать в Linux. Например, если вы создали раздел для /usr (смотрите Раздел 2.2.3), вы сможете описать имя и куда примонтировать раздел (скажем, на /usr или `/usr/bin`).

! **Предупреждение!** Создание файловой системы в разделе разрушает данные, находившиеся на этом месте. Убедитесь, что вы не уничтожаете данные, которые следует сохранить. Даже если вы уже создали файловую систему, используя `mke2fs` (смотрите Раздел 2.3.5), вы должны использовать пункт меню `Target` (цель) для описания разделов, в которые будет инсталлирован Linux.

Source

Этот пункт меню позволяет описать, с чего будет инсталлироваться Slackware: с дискеты, жесткого диска или CD-ROM. Если вы инсталлируете с жесткого диска, вам будет задан вопрос, какой раздел выделяется файлам Slackware и какого он типа. Например, Если у вас Slackware-файлы в разделе MS-DOS, введите имя раздела (скажем, `/dev/hda1`) и выберите MS-DOS FAT в качестве типа. Далее вам будет задан вопрос, в каком каталоге данного раздела можно найти эти файлы. Например, у вас Slackware-файлы, размещенные в каталоге `C:`
`SLACK` в вашем разделе MS-DOS, введите

```
/slack
```

как местоположение. Обратите внимание, что здесь обычный слэш "/". Если вы инсталлируете с CD-ROM, вас спросят тип используемого устройства CD-ROM, а также, где на CD-ROM находятся файлы. Многие CD-ROM содержат файлы в каталоге `/slakware`, но это зависит от версии. Если вы инсталлируете Slackware Professional (Slackware Professional - версия Slackware, поставляемая Morse Telecommunications), то используются два каталога CD-ROM. `slakware` используется для стандартной системы, которая инсталлирует файлы прямо на ваш жесткий диск. `slackpro` используется на базирующейся на CD-ROM системе, где многие файлы доступны прямо с CD-ROM. Это позволяет сэкономить дисковое пространство, но доступ ко многим файлам заметно медленнее. Некоторые другие поставщики Slackware предоставляют возможности работы прямо с CD-ROM. Но, если у вас есть свободное место на диске, мы не рекомендуем работать со Slackware прямо с CD-ROM. Это медленно.

При инсталляции может быть сообщение Slackware об ошибке монтирования. Это часто говорит о проблеме доступа к жесткому диску или CD-ROM. Если вы увидите такие сообщения об ошибках, посмотрите дополнительную информацию в Разделе 2.5.3.

Disk sets

Эта опция меню позволяет выбрать дисковые наборы, которые вы будете инсталлировать. Как минимум, вы должны инсталлировать дисковый набор `A`.

Для выбора дисковых наборов просто используйте стрелки и клавишу пробела. Обратим внимание, что выбор конкретного дискового набора не означает, что все пакеты диска будут инсталлированы; перед инсталляцией пакетов вам будут выдаваться комментарии "optional" или "recommended."

Install

Наконец, этот пункт меню непосредственно инсталлирует программы в вашу систему. Можно при этом следовать подсказкам. Большинство пользователей выбирает режим "normal." Для каждого выбранного дискового набора выбираются "нужные" пакеты для инсталляции, что сопровождается подсказками. Если вы инсталлируете с дискеты, то будут выдаваться также сообщения о необходимости вставить следующую дискету.

После инсталляции каждого пакета выдается краткое сообщение. Но если у вас нет предварительного опыта работы с Linux или UNIX, многие из этих сообщений будут для вас маловразумительными. Важно, что соответствующий пакет инсталлирован, так что можете не ломать голову над всем, что при этом система решила вам сообщить.

Наиболее типичная ошибка, с которой здесь можно столкнуться - на дискете не обнаруживается нужный файл или ошибка возникает при чтении с дискеты. Последнее сообщение может свидетельствовать о том, что файлы на дискете попорчены или неполны. Любые дискеты, порождающие эти сообщения, должны быть заменены, и вам следует заново инсталлировать дисковые наборы, содержащиеся на этих дискетах. Смотрите также Раздел 2.5.3.

Вы можете также столкнуться с сообщениями об ошибках, при попытке обращения к CD-ROM; убедитесь, что CD-ROM чистый, нет следов от пальцев и т.п.

Configure

Этот пункт меню выполняет пост-инсталляционное конфигурирование системы. Это описывается в следующем разделе.

2.3.7 Создание загрузочной дискеты или инсталляция LILO

Каждый дистрибутив представляет какие-то средства для загрузки вашего нового Linux после инсталляции. Во многих случаях инсталляционная процедура создаст загрузочную дискету, содержащую ядро Linux, конфигурированное для использования вновь созданной файловой системы. Для того, чтобы загрузить Linux, вы должны загрузиться с этой дискеты, и управление после этого будет передано жесткому диску. В других дистрибутивах эта загрузочная дискета одновременно является и инсталляционной дискетой.

Многие дистрибутивы дают возможность инсталлировать **LILO** на ваш жесткий диск. LILO - это программа, которая размещается в главной загрузочной записи (master boot record) диска. Она может загружать ряд операционных систем, включая MS-DOS и Linux, и позволяет в момент загрузки выбирать, что именно загружать.

В дистрибутиве Slackware пункт меню `Configure` в `setup` позволяет создавать загрузочную дискету, как и инсталлировать LILO. Эти опции должны комментировать свои действия. Пункт меню `Configure` позволяет также описывать ваши модем, мышь и информацию о временной зоне.

Чтобы LILO успешно инсталлировалась, необходимо многое знать о конфигурации системы, например, какой раздел какую операционную систему содержит, как загружать каждую из систем и т.д. Многие дистрибутивы при инсталляции LILO пытаются "угадать" соответствующие параметры конфигурации вашей системы. Но в некоторых дистрибутивах автоматическая инсталляция LILO может потерпеть неудачу и оставить вашу главную загрузочную запись в "подвешенном" состоянии (хотя маловероятно, что при этом будет причинен ущерб данным на диске). Особенно, если вы применяете Boot Manager операционной системы OS/2, вы *не должны* пользоваться автоматической процедурой инсталляции LILO. Существуют специальные инструкции по использованию LILO совместно с Boot Manager, которые будут рассмотрены позже.

Во многих случаях лучше использовать загрузочную дискету, пока у вас не появится возможность самому конфигурировать LILO вручную. Если вы обладаете повышенной доверчивостью, вы можете пользоваться автоматической инсталляцией LILO, если она есть в вашем дистрибутиве.

В Главе 4 мы обсудим в деталях, как конфигурировать и инсталлировать LILO конкретно для вашего setup.

Если все идет хорошо, примите наши поздравления! Вы аккуратно инсталлировали Linux на вашей системе. Выпейте диетической Коки или чего-нибудь другого - вы заслужили.

В случае, если вы напоролись на неприятности, следующий раздел опишет наиболее характерные глюки при инсталляции Linux, и как с ними бороться.

2.3.8 Дополнительные процедуры инсталляции

Некоторые дистрибутивы Linux снабжены рядом дополнительных инсталляционных процедур, позволяющих конфигурировать различные пакеты, такие как TCP/IP, X

Window System и т.д. Если у вас есть эти конфигурационные опции периода инсталляции, вам может быть будет интересно предварительно прочитать в этой книге об особенностях конфигурации этих программ. Иначе вам следует отложить эти процедуры до тех пор, когда вы не придете к полному пониманию, как конфигурировать программы.

Как хотите, если все идет наперекосяк - плывите по течению и смотрите, что из всего этого получится. Весьма сомнительно, что все, что вы натворите сейчас, нельзя будет переделать потом (впрочем, постучите по дереву).

2.4 Постинсталляционные процедуры

После того, как вы закончите инсталляцию Linux, мало что остается сделать перед тем, как начать использовать систему. В большинстве случаев вы можете перезагрузить систему, войти под root и начать эксплуатировать систему. (Все дистрибутивы имеют слегка различающиеся приемы, реализующие то же самое).

Сейчас самое подходящее время рассказать, как перезагружаться и выключать систему в процессе эксплуатации. Ни в коем случае не перезагружайте и не выключайте систему путем нажатия "reset" или доброго старого способа "заткнуть вулкан" - нажать сразу $ctrl - alt - del$.[13] Не следует также "бесхитростно" вырубать питание. Как и в большинстве систем UNIX, Linux хранит записываемую информацию в кэше оперативной памяти. Поэтому, если вы внезапно перезагрузитесь без "чистого" закрытия системы, вы можете попортить данные на диске, что может привести к невосполнимым потерям.

Самый простой способ выключить систему - использовать команду `shutdown`. Например, для немедленного выключения и перезагрузки используйте следующую команду (в `root`):

```
# shutdown -r now
```

Эта команда чисто перезагрузит вашу систему. Руководство на команду `shutdown` описывает и другие возможные аргументы командной строки.[14]

Обратите внимание, что многие дистрибутивы Linux не предлагают команду shutdown на средствах инсталляции. Это означает, что при первой загрузке после инсталляции,

---

[13] Правда на большинстве Linux систем комбинация $ctrl - alt - del$ приведет к нормальному выключению через команду shutdown.

[14] Можно посмотреть возможности команды, обратившись к машинному руководству `man shutdown`.

вам может потребоваться для останова $ctrl-alt-del$. А после этого вы всегда должны использовать команду `shutdown`.[15]

После того, как вы получили возможность использовать систему, осталось еще несколько акций связанных с конфигурированием, которые следовало бы предпринять. Первое, это создать себе пользовательский account (и, возможно, для других пользователей, которые будут иметь доступ к (в) этой системе) прим.переводчика : завести*account* – значитзарегистрироватьсявсистеме. Создание пользовательских accounts описано в Разделе 4.4. Обычно все, что вы должны сделать, это войти под именем root и выполнить команду `adduser` (иногда `useradd`). При регистрации будет несколько подсказок, которые помогут вам зарегистрировать новых пользователей.

Если вы создали более одной файловой системы для Linux или, если вы используете область своппинга, вам может понадобиться отредактировать файл `/etc/fstab`, чтобы ваши файловые системы были автоматически доступны после загрузки. (Например, если вы используете отдельную файловую систему для `/usr` и не можете обнаружить ни одного из файлов этой файловой системы, может оказаться, что вам просто надо эту систему примонтировать). В Разделе 4.8. описывается эта процедура. Заметим, что дистрибутивы Slackware Linux автоматически конфигурируют ваши файловые системы и области своппинга, так что в описаном выше обычно нет необходимости.

2.5 Борьба с глюками

Практический каждый влипнет в какую-нибудь историю при первой попытке инсталлировать Linux. В большей части случаев это связано с простым неправильным пониманием. прим.переводчика : Тоестьсвязаноспривычкойбыстро, нонеправильносхватыватьмысли. Но иногда может быть кое-что и посерьезнее, как зевок проектировщиков или просто ошибка.

Этот раздел описывает некоторые наиболее часто встречающиеся проблемы инсталляции и как их решать. Если инсталляция прошла успешно, но вы получили неожиданные сообщения об ошибках, они здесь также описываются.

2.5.1 Проблемы загрузки средств инсталляции

Пытаясь первый раз загрузить средства инсталляции, вы можете столкнуться с множеством проблем. Они перечислены ниже. Заметим, что следующие проблемы

---

[15] прим. переводчика: До настоящего момента я закрывал Linux-систему командой `halt`, но `shutdown` тоже работает :-).

не относятся к загрузке вашего вновь инсталлированного Linux. Относительно таких проблем см. Раздел 2.5.4.

- **Ошибка дискеты или средства инсталляции.**

  Наиболее часто встречающийся случай при такого рода проблемах - это запорченная загрузочная дискета. Либо дискета физически повреждена - тогда вы должны восстановить диск, используя исправную дискету, либо испорчены данные на дискете, в этом случае следует проверить правильность перенесения данных на дискету. Во многих случаях вам может помочь простая перезапись дискеты.

  Если вы получили загрузочную дискету по почте или от какого-то другого дистрибутора, вместо самостоятельных попыток восстановить испорченную дискету свяжитесь с дистрибутором и попросите новую загрузочную дискету - но только окончательно убедившись, что именно в дискете причина.

- **Система "зависает" во время или сразу после загрузки.**

  После инсталляции средств загрузки вы увидите номер сообщения ядра, указывающий, какие устройства были распознаны и конфигурированы. После этого обычно выдается "login", позволяющий продолжать инсталляцию (некоторые дистрибутивы вместо этого помещают вас в некоторого рода инсталляционную программу). Система может зависнуть во время этих шагов. Во многих случаях система не зависает, а просто требует много времени на выполнение. Так что, прежде чем решить, что система зависла, убедитесь, что по крайней мере несколько минут дисковод и процессор бездействуют.

  1. После загрузки с помощью LILO, система должна загрузить образ ядра с дискеты. Это может занять несколько секунд; если горит при этом лампочка обращения к дисководу, то это значит, что все идет нормально.
  2. При загрузке ядра SCSI устройства должны быть проверены. Если у вас еще не было инсталлировано какого-нибудь SCSI устройства, система "зависнет" секунд на 15, пока происходит проверка SCSI устройства; обычно это происходит после строки

     ```
     lpinit: lp1 exists (0), using polling driver
     ```

     появившейся на вашем экране.
  3. После окончания загрузки ядра управление передается системе, загружающей файлы с дискеты. Затем вам будет выдана подсказка login или система выйдет в инсталляционную программу. Если вы дошли до подсказки, например, имеющей вид

     ```
     Linux login:
     ```

далее вы должны войти (обычно как root или install - в разных версиях дистрибутивов по-разному). После введения имени пользователя система может задуматься секунд на 20 или более, пока программа инсталляции или shell загружается с дискеты. Опять же лампочка дисковода должна гореть. Так что не думайте, что система опять зависла.

Любой из перечисленных выше пунктов может быть источником проблем. Разумеется, система может и взаправду зависнуть при загрузке, чему может быть несколько причин. Прежде всего, у вас может быть недостаточно памяти (RAM) для загрузки средств инсталляции.

Причина многих системных зависаний - аппаратная несовместимость. Раздел 1.8 последней главы представляет обзор поддерживаемого ОС Linux оборудования. Даже если ваша аппаратура поддерживается, у вас могут быть проблемы, связанные с несовместимостью конфигурации оборудования, которые тоже могут быть причиной зависания. Смотрите Раздел 2.5.2, где обсуждаются вопросы аппаратной несовместимости.

- **Системные сообщения об ошибках памяти в процессе инсталляции.**

  Этот пункт относится к количеству памяти, которая имеется в вашем распоряжении. На системе с 4M RAM или менее у вас могут быть проблемы с самой загрузкой средств инсталляции. Это потому, что многие дистрибутивы используют “ramdisk”, которая является файловой системой, загружаемой прямо в RAM во время операций, использующих средства инсталляции. Полный образ инсталляционной дискеты, например, может быть загружен на ramdisk, что может потребовать более мегабайта памяти.

  Решение этой проблемы - подготовить опцию ramdisk при загрузке средств инсталляции. Каждая версия имеет процедуры реализации этого; в версии SLS, например, вы печатаете “`floppy`”, когда появится подсказка LILO при загрузке диска ”a1”. Детали посмотрите в документации на дистрибутив.

  Вы можете не увидеть сообщение “out of memory” при попытке загрузиться или инсталлировать программы; вместо этого система может неожиданно зависнуть или сорвать загрузку. Если система зависла и никакие предыдущие об’яснения не помогают, попытайтесь отключить (disable) ramdisk.

  Имейте в виду, что Linux сам по себе требует не менее 2 M RAM для минимального функционирования; некоторые версии требуют наличия 4M и даже более.

- **Система сообщает об ошибках, таких как “permission denied” (обращение запрещено) или “file not found”(файл не найден) в процессе загрузки.**

Это говорит о том, что средства инсталляции неисправны. Если вы попытаетесь загрузиться со средств инсталляции (и вы уверены, что все делаете правильно), то у вас не должно появляться сообщений, вроде вышеупомянутых. Свяжитесь с дистрибутором вашего Linux и обсудите с ним проблему. Может быть нужна новая копия. Если вы переписали загрузочный диск сами, попробуйте пересоздать этот загрузочный диск, может это решит проблему.

- **Система при загрузке выдает сообщение "VFS: Unable to mount root".**

  Это сообщение об ошибке означает, что корневая файловая система (сама находящаяся на средстве загрузки) не может быть найдена. То ли ваши средства загрузки каким-то образом испорчены, то ли вы неправильно пытаетесь загружать систему.

  Например, многие дистрибутивы на CD-ROM требуют, чтобы при загрузке лазерный диск находился в дисководе. Убедитесь также, что дисковод CD-ROM включен и что-то делает. Подробнее смотрите в Разделе 2.5.2.

  Если вы уверены, что вы правильно загружаете систему, то возможно неправильно работают ваши средства загрузки. Но это нетипичная ситуация, попробуйте другие варианты, прежде чем пытаться использовать другую загрузочную дискету или ленту.

2.5.2 Аппаратные проблемы

Наиболее общий случай, когда инсталляция или использование Linux приходят в противоречие с аппаратурой. Даже если вся ваша аппаратура поддерживается Linux, неправильное конфигурирование или конфликты между отдельными устройствами могут иногда приводить к странным результатам: устройства могут не распознаваться на этапе загрузки или система может зависать.

Важно локализовать эти аппаратные проблемы, если вы подозреваете, что именно они являются источником ваших неприятностей. В последующих разделах мы опишем некоторые общие проблемы, связанные с аппаратурой, и как решать их.

Локализация аппаратных проблем Если вы столкнулись с проблемой, которая по вашему мнению носит аппаратный характер, первое, что вы должны сделать, это попытаться локализовать проблему. Это означает, что исключая все возможные составляющие и (обычно) саму операционную систему, вы постепенно шаг за шагом выделяете неисправную часть аппаратуры.

Это не так тяжело, как иногда может казаться. Первоначально вы должны отключить от системы все несущественное оборудование, а затем определить, какое устройство в действительности является источником неприятностей, подключая шаг за шагом устройства. Это означает, что вы должны отключить все устройства кроме контроллеров гибкого диска и видео, а также, разумеется, клавиатуры. Даже такие невинные на первый взгляд устройства, как мышь, могут внести большую сумятицу в ваши мозги.

Например, предположим, что система зависла во время загрузки при распознавании платы Ethernet. Вы можете предположить, что имеет место конфликт или это проблема данной платы Ethernet. Простой и быстрый способ определиться - это вытащить плату Ethernet и попытаться вновь загрузиться. Если все пойдет нормально, то (a) плата Ethernet не поддерживается Linux (смотри Раздел 1.8 относительно совместимых плат), или (b) существует адрес или IRQ, конфликтующие с платой.

“Конфликт адреса или IRQ ?” А это-то, скажите на милость, что еще может значить? Все устройства в вашей машине используют IRQ прим.переводчика : $IRQ - InterruptReQuest$ или линию запросов прерывания, чтобы сообщить системе, что система должна для них что-то сделать. Вы можете представить себе IRQ как веревочку, за которую устройство дергает, когда ему надо, чтобы система позаботилась о выполнении какого-то поступившего запроса. Если более, чем одно устройство дергает за эту веревочку, ядро не способно определить, какое устройство нуждается в обслуживании. Вот вам и глюк.

Поэтому убедитесь, что все инсталлированные вами устройства используют уникальные линии IRQ. В общем случае IRQ для устройства может быть установлен с помощью переключения джамперов (jumpers) на плате; детали смотрите в документации на конкретное устройство. Некоторые устройства вообще не используют IRQ, но предполагается, что вы конфигурировали их, так, что они смогут им воспользоваться. (Хорошие примеры тому контроллеры Seagate ST01 и ST02 SCSI).

В некоторых случаях ядро, находящееся на ваших средствах инсталляции, конфигурируется для использования конкретного IRQ для конкретного устройства. Например, в некоторых дистрибутивах Linux ядро предварительно сконфигурировано так, чтобы использовать IRQ 5 для контроллера TMC-950 SCSI, контроллера CD-ROM Mitsumi и драйвер мыши busmouse. Если вы хотите использовать два или более из этих устройств, вам необходимо будет вначале инсталлировать Linux только с одним из этих устройств, подключенным к системе, а затем перекомпилировать ядро, чтобы сменить IRQ, выделенное для другого из них по умолчанию. (Смотри Главу 4 по поводу перекомпиляции ядра).

Другая область, где могут возникнуть конфликты аппаратуры - это каналы DMA (Direct Memory Access) (каналы прямого доступа к памяти), адреса ввода-вывода (I/O) и адреса разделяемой памяти (shared memory addresses). Все вышеперечисленное есть механизмы, через которые система взаимодействует с различными устройствами. Некоторые платы Ethernet, например, используют разделяемую память также как и IRQ в качестве интерфейса с системой. И если они конфликтуют с другими драйверами - система ведет себя непредсказуемо. Вы должны быть готовы к изменению канала DMA, адресов ввода-вывода или разделяемой памяти для различных устройств с помощью переустановки джамперов. (К сожалению, некоторые устройства не позволяют сделать такие переустановки).

Документация на различные устройства должна указывать IRQ, канал DMA, адрес ввода-вывода или адрес разделяемой памяти, которые используют устройства, и как их конфигурировать. И опять, простейший способ справиться с этой проблемой, это просто временно отключить конфликтующие устройства до того, как вы определите причину конфликта.

Таблица 2.2 представляет перечень IRQ и каналов DMA, используемых различными ”стандартными” устройствами, стоящими во многих системах. Практически все системы имеют эти устройства, так что вам следует избегать установок IRQ и DMA других устройств на эти значения.

Device	I/O-адрес	IRQ	DMA
ttyS0 (COM1)	3f8	4	n/a
ttyS1 (COM2)	2f8	3	n/a
ttyS2 (COM3)	3e8	4	n/a
ttyS3 (COM4)	2e8	3	n/a
lp0 (LPT1)	378 - 37f	7	n/a
lp1 (LPT2)	278 - 27f	5	n/a
fd0, fd1 (floppies 1 and 2)	3f0 - 3f7	6	2
fd2, fd3 (floppies 3 and 4)	370 - 377	10	3

Таблица 2.2: Обычные установки для устройств

Проблемы распознавания жесткого диска или контроллера При загрузке Linux вы увидите серии посланий, выдаваемых на экран, вроде:

```
Console: colour EGA+ 80x25, 8 virtual consoles
Serial driver version 3.96 with no serial options enabled
tty00 at 0x03f8 (irq = 4) is a 16450
tty03 at 0x02e8 (irq = 3) is a 16550A
lpinit: lp1 exists (0), using polling driver
```

```
...
```

Здесь ядро распознает различные устройства, имеющиеся в системе. В некоторый момент вы увидите строчку

```
Partition check:
```

(Проверка раздела), за которой следует список распознанных разделов, например:

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

Если по какой-то причине ваши дисководы или разделы нераспознаны, вы никаким образом не сможете к ним добраться.

Это может произойти по нескольким причинам:

- **Жесткий диск или контроллер не поддерживается.** Если вы используете контроллер жесткого диска (IDE, SCSI и тому подобные), из тех, которые не поддерживаются в Linux, ядро не распознает ваш раздел на этапе загрузки.
- **Жесткий диск или контроллер неправильно конфигурированы.** Даже если ваш контроллер поддерживается в Linux, он может быть неправильно конфигурирован. (Особенно эта проблема характерна для контроллеров SCSI; большинство не-SCSI контроллеров будет хорошо работать без дополнительной конфигурации).

  Для решения такого рода проблем обращайтесь к соответствующей документации на жесткие диски и/или контроллеры. В частности, многие жесткие диски потребуют переустановки джамперов, если они будут использоваться в режиме "подчиненного" (“slave”) драйвера (например, в качестве второго жесткого диска). Самый железный способ проверить наличие такой ситуации - это загрузить MS-DOS или еще какую-нибудь другую операционную систему, которая заведомо должна работать с этим жестким диском и контроллером. Если вы получите доступ к диску и контроллеру из другой операционной системы, то значит ваши проблемы не в конфигурировании аппаратуры.

  Смотрите Раздел 2.5.2.1 (ранее) по поводу разрешения возможных конфликтов устройств и Раздел 2.5.2.3 (далее) по поводу конфигурирования SCSI-устройств.
- **Контроллер конфигурирован правильно, но не распознается.** Некоторые без-BIOS-ные SCSI-контроллеры требуют от пользователя описания контроллера

на этапе загрузки. В Разделе 2.5.2.3 (далее) описывается, как осуществить определение этих контроллеров.

- **Не распознается геометрия жесткого диска.** Некоторые системы, такие, как IBM PS/ValuePoint, не помещают информацию о геометрии жесткого диска в память CMOS, где Linux ожидает ее найти. Также, некоторым SCSI-контроллерам надо сообщать, где найти геометрию диска, чтобы Linux мог распознать формат вашего диска.

  Многие дистрибутивы имеют загрузочную опцию для описания геометрии диска. В общем случае, при загрузке средств инсталляции, вы можете описать геометрию драйвера в ответ на подсказку загрузчика LILO с помощью команды, например:

  ```
  boot:  linux hd=cylinders,heads,sectors
  ```

  где <cylinders>, <heads> и <sectors> соответсвуют числу цилиндров, головок и секторов на трек у вашего диска.

После инсталляции Linux вы будете иметь возможность инсталлировать LILO, который позволит вам загружаться с жесткого диска. В это время вы можете описать геометрию для инсталляционной процедуры LILO, что позволит не вводить геометрию при каждой загрузке. Более подробно о LILO смотрите в Главе 4.

Проблемы со SCSI-контроллерами и устройствами Здесь описываются некоторые из наиболее типичных проблем, возникающих со SCSI-контроллерами и устройствами, такими, например, как CD-ROM, жесткие диски и ленты. Если у вас проблемы заставить Linux распознавать диск или контроллер, читайте дальше.

Linux *SCSI HOWTO* (см. Приложение A) содержит много полезной информации о таких SCSI-устройствах, в дополнение к перечисленным здесь. Иногда требуется почти акробатическая ловкость при конфигурировании SCSI.

- **SCSI-сказки устройство распознается всеми возможными идентификаторами (ID).** Это связано с привязкой устройств к одному и тому же адресу с контроллером. Вам следует изменить установку переключателей так, чтобы драйвер и контроллер использовали различные адреса.

- **Linux сообщает об обнаруживаемых ошибках, хотя известно, что устройство работает безошибочно.** Это может происходить из-за плохого кабеля или плохого раз'ема. Если ваша SCSI-шина не имеет надежных контактов с обеих сторон - может возникать ошибка доступа к SCSI-устройствам. Если у вас возникают сомнения, всегда проверяйте кабель.

- **SCSI-устройства сообщают об ошибках истечения времени.** Это обычно происходит из-за конфликтов IRQ, адресов DMA или устройств. Следует проверить также, что прерывания вашим контроллером обрабатываются корректно.

- **SCSI-контроллеры, использующие BIOS не идентифицируются.** Распознавание контроллеров, использующих BIOS, потерпит неудачу, если BIOS отключен или "подпись" вашего контроллера не распознается ядром. Дополнительную информацию можно найти в Linux *SCSI HOWTO*.

- **Контроллеры, использующие отображаемый в память ввод-вывод, не работают.** Это происходит, когда порты отображаемого в памяти ввода-вывода буферизируются некорректно. Или определите в установках XCMOS адресное пространство контроллера, как некэшируемое, или отключите также и кэш.

- **При разбиении на разделы будет выдано сообщение, что "cylinders 1024" или что вы не сможете загрузиться из раздела, имеющего номера цилиндров более 1023.** BIOS ограничивает число цилиндров числом 1024 и любой раздел, использующий большие номера цилиндров, будет неприемлем с точки зрения BIOS. Применительно к Linux это касается только загрузки; после того, как система загружена, вы сможете обращаться к разделу. Вы можете выбирать, загружать ли Linux с дискеты или из раздела, использующего цилиндры с номерами меньше 1024. Относительно создания загрузочной дискеты или инсталляции LILO смотрите Раздел 2.3.7.

- **CD-ROM или другие устройства, которые могут дополнительно вставляться (удаляться) в компьютер, не распознаются на этапе загрузки.** Постарайтесь загрузиться с подключенным CD-ROM (или диском). Для некоторых устройств это необходимо.

Если ваш SCSI-контроллер нераспознан, возможно вам следует инициировать (force) распознавание аппаратуры на этапе загрузки. Это особенно важно для без-BIOS-ных SCSI-контроллеров. Большинство дистрибутивов позволяет описывать IRQ контроллеров и адресов разделяемой памяти во время загрузки средств инсталляции. Например, если вы используете контроллер TMC-8xx, вы можете ввести

```
boot:  linux tmx8xx=interrupt,memory-address
```

в ответ на подсказку загрузчика LILO, где <interrupt> - IRQ контроллера и <memory-address> - адрес разделяемой памяти. Сможете ли вы это сделать, зависит от используемого вами дистрибутива Linux, так что относительно деталей посмотрите документацию.

2.5.3 Проблемы инсталляции программ

Предполагается, что инсталляция программ Linux должна проходить без особых хлопот, если вы счастливый человек. Единственные проблемы, с которыми вы можете столкнуться, это испорченные средства инсталляции или отсутствие достаточного места на файловой системе Linux. Вот перечень наиболее характерных проблем:

- **Системные сообщения “Read error”(ошибка чтения), “file not found”(не найден файл) или другие ошибки во время попытки инсталлировать программы.** Это говорит о проблемах с вашими средствами инсталляции. Если вы инсталлируете с дискеты, имейте в виду, что дискеты очень склонны к такого рода недостаткам. Убедитесь, что вы используете новые исправные и свежеотформатированные дискеты. Если у вас есть на диске разделы MS-DOS, многие дистрибутивы Linux позволят вам инсталлировать с жесткого диска. Это может быть быстрее и более надежно, чем использование дискет.

  Если вы используете CD-ROM, убедитесь в отсутствии на нем царапин, пыли или других гадостей, которые могут приводить к ошибкам. Причиной может быть и то, что соответствующее средство инсталляции имеет неподходящий формат. Например, при использовании дискет многие дистрибутивы Linux требуют, чтобы дискета была отформатирована в формате high-density MS-DOS. (Загрузочная дискета - исключение; в большинстве случаев она вообще не в формате MS-DOS). Если все прочее потерпело неудачу, либо достаньте новый набор дискет с дистрибутивом или перепишите его на новые дискеты, если вы скачали дистрибутив откуда-то.

- **Системные сообщения вроде “tar: read error” (tar: ошибка чтения) или “gzip: not in gzip format”(gzip: не в формате gzip).** Часто это связано с испорченными файлами на средствах инсталляции. Другими словами, ваши дискеты могут быть нормальными, но вот данные на них каким-то образом испорчены. Например, вы каким-то образом скачали программы Linux, используя текстовый (а не бинарный) режим, тогда ваши файлы уж точно будут негодными для инсталляции.

- **Системные сообщения об ошибках, такие как “device full” (устройство заполнено) в процессе инсталляции.** Это верный признак того, что вы вышли за пределы отведенного пространства при инсталляции. Не все дистрибутивы способны с этим разобраться; вы не сможете прервать инсталляцию и вынуждены дождаться, когда система сама остановится.

Обычное решение в этой ситуации - пересоздание файловой системы (с помощью команды `mke2fs`), которая удаляет частично инсталлированные программы. Вы далее можете попытаться переинсталлировать программы, выбирая на этот раз меньшее количество программ, подлежащих инсталляции. В других случаях вам может понадобиться начать с полного удаления и перераспределения разделов и размеров файловой системы.

- **Системные сообщения об ошибках, такие как "read_intr: 0x10" при обращении к жесткому диску.** Это обычно говорит о наличии плохих блоков на диске. Однако, если вы получили это сообщение во время выполнения mkswap или mke2fs, причиной этого могло быть то, что система имела проблемы с доступом к вашему диску. Это может быть как проблема аппаратуры (см. Раздел 2.5.2), так и результат неправильного описания геометрии. Если вы применяли

  ```
  hd=cylinders,heads,sectors
  ```

  опцию периода загрузки, чтобы инициировать определение геометрии своего жесткого диска и описали геометрию некорректно, то вы должны будете познакомиться с этой проблемой. Это также может случиться, если геометрия вашего драйвера описана некорректно в CMOS.

- **Системные сообщения об ошибках , вроде "file not found" или "permission denied".** Это может случиться, если не все необходимые файлы представлены на средствах инсталляции (смотрите следующий раздел) или существует проблема разрешения доступа. Например, про некоторые дистрибутивы Linux известно, что они сами по себе содержат ошибки. Это обычно обнаруживается очень быстро, да и случается не часто. Если вы подозреваете, что программы дистрибутива содержат ошибки и уверены, что вы ничего не сделали неправильно, свяжитесь с сопровождающими дистрибутив и сообщите об ошибке.

Если у вас появляются другие странные ошибки во время инсталляции Linux (особенно если вы сами переписали где-то эти программы), убедитесь, что вы действительно списали все необходимое. Например, некоторые используют команду FTP

```
mget *.*
```

для скачивания программ Linux через FTP. Она скачает только те файлы, которые содержат "." в именах файлов; если есть файлы без ".", вы их не получите. В этом случае уместной была бы команда

```
mget *
```

Самый лучший совет - заново пересмотреть все шаги, которые вы совершили, если у вас застопорилось дело. Вы можете наивно думать, что вы все делали правильно, когда на самом деле вы забыли сделать какой-то маленький, но важный шаг, где-то посреди нелегкого пути инсталляции. Во многих случаях даже сама попытка заново переписать или заново инсталлировать Linux может натолкнуть на решение действительной проблемы. Не надо биться головой об стену дольше, чем надо!

Кроме прочего, если Linux завис при инсталляции, причины могут быть в аппаратуре. Смотрите по этому поводу Раздел 2.5.2.

2.5.4 Проблемы после инсталляции Linux

Вы потратили целых полдня, инсталлируя Linux. Чтобы выделить под него место, вы стерли свои разделы с MS-DOS и OS/2 и не без утирания слез стерли свои копии "SimCity" и "Wing Commander"... Вы перезагрузили систему, а ничего не произошло. Или еще хуже того, *что-то* произошло, но не то, что должно было произойти. Ну и что делать?

В Разделе 2.5.1 мы обсуждали некоторые из наиболее типичных проблем, возникающих при загрузке Linux со средств инсталляции. Многие из этих проблем могут быть и здесь. В довершение ко всему вы можете стать жертвой одной из следующих напастей.

Проблемы загрузки Linux с дискеты Если вы используете дискеты для загрузки Linux, вам может потребоваться описать местоположение вашего корневого раздела linux во время загрузки. Это обычно случается, когда вы используете исходную инсталляционную дискету, а не специальную загрузочную дискету, созданную в процессе инсталляции.

При загрузке дискеты, надо держать *shift* или *ctrl*. Это приведет вас к загрузочному меню; нажмите *tab*, чтобы получить спиок доступных опций. Например, многие дистрибутивы позволяют ввести

```
boot: linux hd=partition
```

где <partition> - имя корневого раздела Linux, например, /dev/hda2. Более детально с вопросом можно познакомиться по документации на дистрибутив.

Проблемы загрузки Linux с жесткого диска Если вам удалось инсталлировать LILO, вместо создания загрузочной дискеты вам следует загружать Linux с жесткого диска. Однако, автоматизированная процедура инсталляции LILO, используемая во многих дистрибутивах, не всегда безупречна.

Она может сделать неправильные предположения относительно формата вашего раздела, в этом случае вы должны будете переинсталлировать LILO, чтобы все стало хорошо. Инсталляция LILO обсуждается в Главе 4.

- **Системные сообщения “Drive not bootable---Please insert system disk.”** (”Устройство незагружаемо---Пожалуйста, вставьте системный диск”). Вы получите такое сообщение об ошибке, если главная загрузочная запись жесткого диска каким-то образом попорчена. Во многих случаях это безопасно и все остальное у вас на диске по-прежнему в порядке. Тут дальше есть несколько путей.

    1. При разбиении диска на разделы с использованием **fdisk** вы могли удалить раздел, который был отмечен как “active”. MS-DOS и другие операционные системы пытаются загрузить такой раздел на этапе загрузки (Linux не обращает внимания на то, является раздел “active” или нет). Вы можете загрузить MS-DOS с дискеты и запустить **FDISK** для установки флага “active” для раздела MS-DOS и все будет хорошо.

       Другая команда, которую можно попробовать (с MS-DOS 5.0 и выше) это

       ```
       FDISK /MBR
       ```

       Эта команда будет пытаться заново сформировать главную загрузочную запись диска для загрузки MS-DOS, переписывая LILO. Если у вас больше нет на жестком диске MS-DOS, вам потребуется загрузить Linux с дискеты и в последующем попытаться инсталлировать LILO.

    2. Если вы создали раздел MS-DOS, используя версию команды **fdisk** из Linux или наоборот, это может быть причиной ошибки. Вам следует создавать разделы для MS-DOS, используя только версии **FDISK** для MS-DOS. (Это относится и к другим операционным системам, которые существуют наряду с MS-DOS). Здесь лучшее решение - либо начать с того, что все стереть и переразбить диск правильно, либо удалить и пересоздать плохие разделы, используя исправные версии **fdisk**.

    3. Инсталляционная процедура LILO может потерпеть неудачу. В этом случае вам следует либо загрузиться с загрузочной дискеты для Linux (если она у вас есть) или с исходного средства инсталляции. В любом случае вы будете иметь возможность для описании корневого раздела Linux, который

будет использован при загрузке. Нажмите *shift* или *ctrl* во время загрузки и нажмите *tab* из меню загрузки, чтобы получить список опций.

- **При загрузке системы с жесткого диска MS-DOS (или другая из существующих операционных систем) стартует вместо Linux.** Прежде всего убедитесь, что вы действительно инсталлировали LILO при инсталляции программ Linux. Если это оказалось не так, система будет загружать MS-DOS (или какую-нибудь другую операционную систему из собранных вами), когда вы пытаетесь загрузиться с жесткого диска. Для того, чтобы загрузить Linux с жесткого диска, вам необходимо инсталлировать на жеский диск LILO (см. Главу 4).

  С другой стороны, если вы *все-таки инсталлировали* LILO, но другая операционная система загружается вместо Linux, то необходимо конфигурировать LILO так, чтобы она загружала другие операционные системы по умолчанию. Во время загрузки системы держите нажатой клавишу *shift* или *ctrl*, а затем нажмите *tab* в ответ на подсказку загрузчика. В результате вы получите список операционных систем, которые можно загрузить. Выберите соответствующую опцию (часто просто ``linux''), чтобы загрузить Linux.

  Если вы хотите сделать Linux системой, загружаемой по умолчанию, вам необходимо реинсталлировать LILO. Смотрите Главу 4.

  Возможно также, что вы пытались инсталлировать LILO, но инсталляция потерпела неудачу. Смотрите предыдущий пункт.

Проблемы входа в систему После загрузки Linux вам (на экран) должна быть выдана подсказка вроде этой:

```
linux login:
```

В этот момент либо документация на дистрибутив, либо сама система скажут вам, что делать дальше. В большинстве дистрибутивов вы просто войдете в систему под именем `root` (суперпользователь, администратор) без пароля. Другие возможные имена для входа `guest` или `test`.

Большинство новоиспеченных систем Linux не требуют пароля для первоначального входа. Но если система потребует с вас пароль, могут возникнуть проблемы. Прежде всего попробуйте пароль, совпадающий с именем входа; например, если вы вошли как `root`, попробуйте ``root'' в качестве пароля.

Если вы все-таки не можете войти, то это уже проблема. Прежде всего проконсультируйтесь с документацией на дистрибутив. Может быть там где-то закопано правильное имя входа и пароль. Имя входа и пароль могут быть вам сообщены системой во время инсталляции или выведены на экран в виде подсказки.

Причиной этих неприятностей также могут быть проблемы с самой инсталляцией файлов, отвечающих за вход и инициализацию. Если в этом причина, вам может потребоваться переинсталлизация (как минимум части) программ Linux или нужно загрузить ваши средства инсталляции и попытаться решить проблемы "вручную". Смотрите соображения на этот счет в Главе 4.

Проблемы использования системы Если вход в систему прошел успешно, на экран будет выдана подсказка "shell" - командной оболочки (например "`#`" или "`$`") и вы можете немножко поплясать вокруг системы. Но существует ряд проблем, которые могут возникнуть в начале использования системы.

Одна из наиболее типичных начальных проблем, связанных с конфигурированием - установка неверных прав доступа (защиты) файлов и каталогов. Это может выразиться в сообщении

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

которое будет напечатано после входа в систему (на самом деле, всегда, когда вы столкнетесь с сообщением "`permission denied`"("обращение запрещено") вы можете быть с высокой вероятностью уверены, что это проблема защиты файлов).

Во многих случаях это простое дело для команды "chmod", которая может менять права доступа к соответствующим файлам и каталогам. Например, некоторые дистрибутивы Linux использовали (ошибочный) код защиты файлов " 0644 " для корневого каталога (/). А следует использовать команду

```
# chmod 755 /
```

Но, чтобы ввести эту команду, вы должны загрузиться со средства инсталляции и примонтировать вашу корневую файловую систему Linux вручную - заковыристая задача для большинства новичков.

Во время эксплуатации системы вы можете попадать в места, где неверно установлена защита файлов и каталогов или программы работают не так, как конфигурировались. Добро пожаловть в мир Linux! Хотя многие дистрибутивы и не доставляют особых хлопот, лишь немногие из них безупречны. Мы не хотим обсуждать здесь все возможные проблемы. Вместо этого по ходу всей книги мы помогаем вам решать многие проблемы конфигурирования, обучая вас обнаруживать и решать проблемы самостоятельно. В Главе 1 мы детально обсуждали эту философию. В Главе 4 мы даем советы относительно решения многих из типовых проблем конфигурирования.

3 Знакомство с Linux

3.1 Введение

Новые пользователи UNIX и Linux могут быть ошеломлены размерами и очевидной сложностью системы, которая предстала перед ними. Существует много хороших книг по использованию UNIX для всех уровней подготовки: от новичка до эксперта. Но ни одна из этих книг не обсуждает особенности Linux. Хотя 95% всего связанного с использованием Linux абсолютно аналогично другим UNIX-системам, наиболее прямой путь освоения этой системы - это по учебнику, написанному применительно к Linux. Вот эта книга и есть такой учебник.

Эта глава не заводит в дебри деталей и не обсуждает наиболее сложные прим.переводчика : они(иужеунастоже)говорят – продвинутые аспекты Linux. Вместо этого делается попытка поставить новичка крепко на ноги, чтобы он мог в дальнейшем читать и более общие книги по UNIX, понимая базовые различия других UNIX-систем и Linux.

Здесь не предполагается каких-то предварительных знаний, за исключением первоначального знакомства с персональным компьютером и MS-DOS. Но даже если вы не успели побывать пользователем MS-DOS, вы все равно все здесь поймете. На первый взгляд UNIX очень похож на MS-DOS (в конце-концов фрагменты MS-DOS были спроектированы с оглядкой на операционную систему CP/M, которая, в свою очередь, проектировалась с оглядкой на UNIX). Но только при очень уж поверхностном взгляде можно говорить о похожести UNIX и MS-DOS. Если вы абсолютный новичок в мире персональных компьютеров, этот учебник вам поможет.

И прежде, чем начать, призываем: *не бойтесь экспериментировать.* Система вас не укусит. Работая на ней вы ничего не сможете сломать. UNIX имеет встроенные средства защиты, чтобы не дать ”нормальным” пользователям (это теперь и вы) возможность испортить важные для системы файлы. Самое плохое, что вы можете натворить - это уничтожить все свои файлы, а тогда, может быть придется и переинсталлировать заново систему прим.переводчика : какправило, чтобыдовестисистемудопереинсталляции, надоиметьправбольше, чему”нормального”

3.2 Базовые концепции UNIX

UNIX это многозадачная, многопользовательская операционная система. Это означает, что много людей может одновременно использовать один компьютер, выполняя много различных задач. (Это существенное отличие от MS-DOS, где только один человек может использовать в данный момент операционную систему). В UNIX пользователи должны себя идентифицировать при входе, что состоит из двух шагов: **ввода имени** (имя, по которому вас идентифицирует система) и **входной пароль**, который является вашим секретным словом для открытия вашего счета (регистрации в системе). Поскольку только вы знаете пароль, никто не может войти в систему под вашим именем.

В традиционных UNIX-системах системный администратор присвоит вам имя и начальный пароль при вашей регистрации в системе (при заведении в системе нового пользователя). Но поскольку на своем персональном компьютере вы и системный администратор, вы должны себя (как пользователя) зарегистрировать в системе, прежде чем в нее войдете (смотрите Раздел 3.2.1 ниже). Для дальнейших разговоров возьмем условное имя “`larry`”.

Кроме прочего, каждая система UNIX имеет приписанное ей **hostname** (хозяйское имя). Это хозяйское имя добавляет машине характера и очарования. Hostname используется для идентификации отдельных машин в сети, но даже если ваша машина не в сети, она все равно должна иметь hostname. В Разделе 4.10.2 мы подробно расскажем об установке hostname на вашей машине. Например, имя машины, обсуждаемой ниже - “`mousehouse`” (мышинная норка).

3.2.1 Регистрация в системе (открытие счета)

Прежде, чем вы сможете использовать систему, вы должны зарегистрировать себя в системе. Это необходимо потому, что неразумно использовать имя суперпользователя (`root`) для обычных нужд. Пользователь root нужен для выполнения привилегированных команд и сопровождения системы, как это описывается в Разделе 4.1.

Для того, чтобы зарегистрировать себя, вам необходимо зайти в систему под именем `root` и использовать команду `useradd` или `adduser`. Об этой процедуре смотрите подробней в Разделе 4.4.

3.2.2 Вход в систему

При входе вы увидите на экране подсказку, например, такого вида:

```
mousehouse login:
```

Введите свое имя и нажмите клавишу *Return*. Наш герой `larry` напечатает следующее:

```
mousehouse login:  larry
Password:
```

Теперь введите ваш пароль (password). При вводе пароль не будет отображаться на экране, так что набирайте внимательнее. Если вы неправильно набрали пароль, то увидите на экране сообщение

```
Login incorrect
```

и вам следует попытаться еще раз.

Когда вы наконец правильно введете имя пользователя и пароль, вы официально будете допущены в систему и можете в ней свободно путешествовать.

3.2.3 Виртуальные консоли

Системная **консоль** - это монитор и клавиатура, связанные непосредственно с системой. (Поскольку UNIX многопользовательская система, вы можете иметь дополнительные терминалы, связанные через последовательные порты с вашей системой, но они не будут консолями). Linux, как и некоторые другие версии UNIX, обеспечивает доступ к **виртуальным консолям** (или VC), которые позволяют войти в систему под несколькими именами в одно время.

Для демонстрации этого войдите в систему (как было показано ранее). Теперь нажмите $alt - F2$. Вы должны снова увидеть подсказку `login:` , то есть перед вами вторая виртуальная консоль, а вы вошли через первую. Чтобы переключиться обратно на первую VC, нажмите $alt - F1$. *Оп-ля!* Вы снова на первой консоли.

Свежеинсталлированный Linux возможно позволит вам работать с четырьмя первыми VC, используя от $alt - F1$ до $alt - F4$. Но возможно обеспечить работу с 12-ю VC - по одной на каждую функциональную клавишу. Как видите, использование VC может быть очень эффективным - вы можете работать на нескольких VC одновременно.

В то время, как использование виртуальных консолей ограничено (кроме прочего, в каждый момент времени вы можете видеть только одну виртуальную консоль) оно дает вам представление о многопользовательских возможностях UNIX. Пока вы работаете на VC #1, вы можете переключиться на VC #2 и начать работу над чем-то другим.

3.2.4 Shells и команды

В большинстве ваших исследований мира UNIX вы будете общаться с ним через оболочку **shell**. Shell - это просто программа, которая воспринимает введенное пользователем, (т.е. команды, которые вы напечатаете) и транслирует это в команды системе. Это можно сравнить с программой `COMMAND.COM` под MS-DOS, которая делает нечто похожее. Shell - это лишь один из интерфейсов UNIX. Существует много различных интерфейсов, таких как X Window System, которая позволяет выполнять команды используя мышь и клавиатуру в сочетании.

Как только вы вошли, система запускает shell и вы можете вводить для него команды. Вот короткий пример. Как раз Larry вошел в систему и система вновь выдала **подсказку**:

```
mousehouse login:  larry
Password:  larry´s password
Welcome to Mousehouse!
/home/larry#
```

“`/home/larry#`” это подсказка shell, показывающая, что он готов принимать команды. (Подробнее про подсказку позже). Давайте попросим систему сделать что-нибудь интересненькое:

```
/home/larry# make love
make:  *** No way to make target `love´.  Stop.
/home/larry#
```

Хм, как оказалось, ”make” - это имя существующей в системе программы и shell пытался выполнить эту команду. (Жаль, но система отнеслась к просьбе недружественно).

Это подводит нас к жгучему вопросу: Что такое команды? Что происходит, когда вы вводите “`make love`”? Первое слово командной строки “`make`” это имя команды, которую предполагается выполнить. Все остальное в командной строке воспринимается как аргументы команды.

Примеры:

```
/home/larry# cp foo bar
```

Здесь имя команды “`cp`”, а аргументы “`foo`” и “`bar`”.

Когда вы вводите команду, shell делает несколько вещей. Во-первых, смотрит на то, что может (должно) быть именем команды и является ли это внутренней для shell командой.

(Внутренняя, это команда, которую shell знает как выполнять. Существует ряд таких команд, мы о них поговорим позже). Shell также проверяет, не является ли команда синонимом другой или требуется подстановка имени. Если этого не надо делать, shell ищет соответствующую этому имени программу на диске. Если shell находит такую программу, он ее выполняет, передавая ей аргументы из командной строки.

В нашем примере shell ищет программу по имени `make` и пытается выполнить ее с аргументом `love`. `make` - это программа, которая часто используется при компиляции больших программ, она берет в качестве аргумента имя "целевого" файла компиляции. В случае ``make love'' мы приказали команде make откомпилировать love. Поскольку make не смог найти файла с таким именем, он сообщил (несколько забавным образом) о невозможности выполнить команду и вернулся в подсказку.

Что случится, если мы введем команду, а shell не сможет найти программу с этой командой? Давайте попробуем:

```
/home/larry# eat dirt
eat:  command not found
/home/larry#
```

Все очень просто, если shell не может найти программу с именем данным в командной строке (здесь ``eat''), он выдает сообщение об ошибке, которое об'ясняет причину невыполнения команды. Вы часто будете видеть это сообщение, если будете вводить имена команд с ошибками. (например, напечатаете ``mkae love'' вместо ``make love'').

3.2.5 Выход из системы

Прежде, чем идти дальше, мы расскажем, как выйти из системы. При наличии подсказки shell используйте команду

```
/home/larry# exit
```

для выхода. Есть другие способы выхода, но этот самый безопасный.

3.2.6 Смена пароля

Вы также должны представлять, как можно менять пароль. Команда "passwd" прим.переводчика : именноспропущеннымибуквамионаипишется спросит вас про старый пароль и про новый. Она попросит дважды ввести новый пороль для надежности. Внимание! Не забывайте свой пароль, иначе вам придется просить системного администратора уничтожить его и установить новый (Если вы и есть системный администратор, смотрите Раздел 4.4).

3.2.7 Файлы и каталоги

Во многих операционных системах (включая UNIX) существует концепция **файла**, по которой его можно рассматривать просто, как набор информации, которому дано имя. Примерами файлов будут: программа, которая может выполняться, письмо, полученное по электронной почте, написанная вами статья. Существенно то, что все, что хранится на диске, хранится в отдельных файлах.

Файлы идентифицируются по именам. Например, файл, содержащий вашу статью может быть сохранен под именем `my-paper`. Эти имена обычно каким-то образом отражают содержание. Не существует стандартного формата имен файлов, как в MS-DOS и других операционных системах; в общем случае имена файлов могут содержать любые символы (кроме `/` - смотрите ниже обсуждение формирования "путей") и ограничены 256 символами по длине.

Одновременно с концепцией файла рассмотрим и концепцию каталога. **Каталог** - это совокупность файлов. Его можно рассматривать как "папку", содержащую множество различных файлов. Каталоги сами по себе также получают имена, по которым вы их различаете. Каталоги организованы в древовидную структуру, т.е. каталоги могут содержать другие каталоги.

К файлу можно обращаться по **пути** (`pathname`), формируемой из имени файла, которому предшествует имя каталога, содержащего файл Например, скажем, Larry имеет каталог, названный `papers`, который содержит три файла: `history-final`, `english-lit`, и `masters-thesis`. (Каждый из этих трех файлов содержит информацию о проводимых Larry работах). Для того, чтобы обратиться к файлу `english-lit`, Larry может указать маршрут:

```
papers/english-lit
```

Как вы видите, имена каталогов и файлов разделяются единичным слэшем (/). Поэтому имена файлов сами по себе не могут содержать этот символ. Пользователи MS-DOS увидят в этом что-то знакомое, поскольку в MS-DOS для этого используется бэкслэш ( ).

Как уже говорилось, каталоги могут быть вставлены друг в друга. Например, пусть Larry в каталоге `papers` имеет другой каталог с названием `notes`. Этот каталог содержит файлы с именами `math-notes` и `cheat-sheet`. Путь файла `cheat-sheet` будет

```
papers/notes/cheat-sheet
```

Поэтому путь - это маршрут, который надо проделать, чтобы добраться до конкретного файла. Каталог выше данного (под)каталога называется **родительским каталогом**. Здесь каталог `papers` является родительским для каталога `notes`.

3.2.8 Дерево каталогов

Большинство систем UNIX имеет стандартную структуру каталогов, что облегчает конкретную установку системы. Структура представляет из себя дерево каталогов, начинающееся с каталога “`/`”, известного под названием ”корневой каталог”. Каталоги ниже `/` относятся к числу важнейших подкаталогов: среди них `/bin`, `/etc`, `/dev`, и `/usr`. Эти каталоги в свою очередь содержат другие каталоги, которые содержат системные конфигурационные файлы, программы и т.д.

В частности, каждый пользователь имеет **домашний каталог**, который выделяется пользователю для хранения его файлов. В вышеприведенном примере все файлы Larry (такие как `cheat-sheet` и `history-final`) содержались в домашнем каталоге Larry. Обычно пользовательский домашний каталог находится под каталогом `/home` и называется именем пользователя. Так домашний каталог Larry будет `/home/larry`.

На Рис. 3.2.8 представлено простое дерево каталогов. Оно даст вам некоторое представление о том, как организуется дерево каталогов в вашей системе.

3.2.9 Текущий рабочий каталог

Команды, которые вы даете shell, выдаются из вашего **текущего каталога**. Вы можете думать о вашем рабочем каталоге, как о каталоге в котором вы находитесь. При начальном входе в систему вашим рабочим каталогом автоматически становится домашний каталог (в нашем случае `/home/larry`). При обращении к файлу вы можете обращаться к нему с учетом вашего местоположения, вместо того, чтобы указывать полный путь.

```
/bin
    |dev
    |etc
    |homelarry
    |        |sam
    |lib
    |proc
    |tmp
    |usrX386
         |bin
         |emacs
         |etc
```

```
|g++-include
|include
|lib
|localbin
|         |emacs
|         |etc
|         |lib
|man
|spool
|srclinux
|tmp
```

Рис 3.1: Типичное (урезанное) дерево каталогов Unix

Вот например, у Larry есть каталог `papers`, а `papers` содержит файл `history-final`. Если Larry хочет посмотреть этот файл, он может использовать команду

```
/home/larry# more /home/larry/papers/history-final
```

Команда more просто показывает файл на экране порциями. Поскольку текущий рабочий каталог Larry `/home/larry`, он вместо этого может обратиться к файлу с учетом своего текущего местоположения. Команда будет

```
/home/larry# more papers/history-final
```

Так что вы можете начинать имя файла (такого как `papers/final`) с символа, отличного от “`/`”, система предполагает, что вы обращаетесь к файлу с учетом вашего текущего рабочего каталога. Это называют **относительным именем** (а полный маршрут - **полное (абсолютное) имя** - т.е. путь от корня до данного имени).

3.2.10 Обращение к домашнему каталогу

Оболочки (shell), т.е. программы, которые читают и выполняют команды пользователя, могут использоваться (одновременно в одной системе) разные. В большинстве систем Linux используются `tcsh` или `bash` при начальной регистрации в системе. В `tcsh` и `bash` вы можете обратиться к своему домашнему каталогу, используя тильду (“ ”). Например, команда

```
/home/larry# more ~/papers/history-final
```

эквивалентна

```
/home/larry# more /home/larry/papers/history-final
```

Символ “ ” просто заменяет имя вашего домашнего каталога.

Вы также можете обратиться к домашнему каталогу другого пользователя с помощью тильды. Имя “ `karl/letters`” будет интерпретировано shell, как “`/home/karl/letters`” (если `/home/karl` - домашний каталог для пользователя karl). Использование тильды упрощает обращение; не существует каталога с именем “ ” - так что это просто ”синтаксический сахар”, который имеется в распоряжении shell.

3.3 Первые шаги в UNIX

Перед тем, как начать, важно заметить, что все имена файлов и команд чувствительны к большим и малым буквам (чего нет в системах типа MS-DOS). Например, команда `make` очень отличается от `Make` или `MAKE`. То же относится и к именам каталогов.

3.3.1 Первая прогулка

Теперь мы можем войти в систему и узнать, как обращаться к файлам и менять местоположение в файловой системе, чтобы упрощать свою жизнь в ней. Команда для перемещения по дереву каталогов - `cd`, (“change directory”). Вы скоро обратите внимание, что многие часто используемые команды Unix состоят из двух-трех букв. Формат команды `cd`:

```
cd directory
```

где <directory> - имя каталога, в который вы желаете перейти. Как мы уже говорили, когда вы входите в систему, вы автоматически оказываетесь в своем домашнем каталоге. Если Larry желает двинуться ниже по дереву, он должен использовать команду

```
/home/larry# cd papers
/home/larry/papers#
```

Как видите, изменилась подсказка, отразив изменение местоположения (новый рабочий каталог). Теперь он в каталоге `papers` и может посмотреть на свой файл `history-final` с помощью команды

```
/home/larry/papers# more history-final
```

Чтобы вернуться назад из подкаталога `papers`, надо использовать команду

```
/home/larry/papers# cd ..
/home/larry#
```

(Обратите внимание на пробел между “`cd`” и “`..`”). Каждый каталог содержит имя “`..`” , которое отсылает к родительскому (для данного каталога) каталогу. Также каждый каталог имеет имя “`.`”, которое ссылается на него самого. Поэтому команда

```
/home/larry/papers# cd .
```

никуда не переведет.

В команде `cd` вы можете использовать маршруты. Чтобы перейти в домашний каталог Карла, вы можете воспользоваться командой

```
/home/larry/papers# cd /home/karl
/home/karl#
```

Используя команду `cd` без аргументов вы из любого места дерева вернетесь в свой домашний каталог.

```
/home/karl# cd
/home/larry#
```

3.3.2 Разглядывание содержимого каталогов

Теперь вы знаете, как ходить-бродить по каталогам, но вероятно возникает вопрос:”Ну и что с того?” Само по себе хождение по каталогам бесполезно, давайте познакомимся с новой командой `ls`. ls (LiSt) выдает на экран перечень файлов и каталогов (по умолчанию из текущего каталога). Например,

```
/home/larry# ls
Mail
letters
papers
/home/larry#
```

Здесь мы видим, что у Larry три ”единицы хранения” в его текущем каталоге: `Mail`, `letters` и `papers`. Но это мало, что говорит: каталоги это или файлы? Можно использовать опцию[16] `-F` в команде `ls`, чтобы получить больше информации.

---

[16] (прим. переводчика: часто в документации по UNIX используют в этом контексте слово ”флаг”)

```
/home/larry# ls -F
Mail/
letters/
papers/
/home/larry#
```

Приписанные справа к именам файлов / говорят о том, что это (под)каталоги.

Использование `ls -F` (обратите внимание "`-F`" пишется без пробела) может дать также "`*`" в конце некоторых имен файлов. Это будет говорить о том, что это **выполняемые** файлы или программы. Если, при вызове `ls -F`, ничего справа не приписано к имени, то это "нормальный" файл, т.е. не каталог и не выполняемый файл.

В общем, каждая команда UNIX может иметь несколько опций в дополнение к другим аргументам. Эти опции обычно записываются со знаком "`-`", как это было показано на примере `ls -F`. Опция `-F` сообщает команде `ls`, что необходимо выдать дополнительную информацию о типе файлов.

Если вы напишете в команде `ls` имя каталога, то она выдаст содержимое указанного каталога.

```
/home/larry# ls -F papers
english-lit
history-final
masters-thesis
notes/
/home/larry#
```

Или, чтобы было интереснее, давайте посмотрим, что имеется в системном каталоге `/etc/`.

```
    /home/larry# ls /etc

Images      ftpusers    lpc         rc.new     shells
adm         getty       magic       rc0.d      startcons
bcheckrc    gettydefs   motd        rc1.d      swapoff
brc         group       mount       rc2.d      swapon
brc~        inet        mtab        rc3.d      syslog.conf
csh.cshrc   init        mtools      rc4.d      syslog.pid
csh.login   init.d      pac         rc5.d      syslogd.reload
default     initrunlvl  passwd      rmt        termcap
disktab     inittab     printcap    rpc        umount
fdprm       inittab.old profile     rpcinfo    update
fstab       issue       psdatabase securetty  utmp
ftpaccess   lilo        rc          services   wtmp
/home/larry#
```

(Для вышедших из MS-DOS пользователей полезно обратить внимание, что имена файлов могут быть длиннее 8 символов и содержать точку на любой позиции. Можно даже использовать несколько точек в одном имени).

Давайте поднимемся вверх по дереву[17] , используя команду “`cd ..`”, а затем спустимся в другой каталог (`/usr/bin`).

```
/home/larry# cd ..
/home# cd ..
/# cd usr
/usr# cd bin
/usr/bin#
```

Вы, разумеется, можете передвигаться по каталогам большими шагами, например, сразу выполнить `cd /usr/bin`.

Постарайтесь погулять по каталогам, используя команды ls и cd. В некоторых случаях вы можете напороться на раздражающее сообщение “`Permission denied`”(обращение запрещено). Это всего лишь сработала система защиты UNIX, чтобы выполнять команды в тех или иных каталогах вы должны иметь на это разрешение. Подробнее об этом поговорим в Разделе 3.9.

3.3.3 Создание новых каталогов

Пора познакомиться с тем, как создавать каталоги. Это связано с использованием команды `mkdir`. Попробуйте следующее:

```
/home/larry# mkdir foo
/home/larry# ls -F
Mail/
foo/
letters/
papers/
/home/larry# cd foo
/home/larry/foo# ls
/home/larry/foo#
```

Наши вам поздравления! Вы только что создали новый каталог и зашли в него. Поскольку пока нет файлов в этом новом каталоге, давайте познакомимся с тем, как копировать файлы.

[17] (прим. переводчика: так уж сложилось, что в UNIX начальной вершиной дерева является ”корень (root)”)

3.3.4 Копирование файлов

Копирование файлов осуществляется командой `cp` (CoPy):

```
/home/larry/foo# cp /etc/termcap  .
/home/larry/foo# cp /etc/shells  .
/home/larry/foo# ls -F
shells    termcap
/home/larry/foo# cp shells bells
/home/larry/foo# ls -F
bells    shells    termcap
/home/larry/foo#
```

Команда `cp` копирует файлы, перечисленные в командной строке, в файл или каталог, указанный последним аргументом.[18] Обратите внимание на то, как мы используем каталог “.”для ссылки на текущий каталог.

3.3.5 Перемещение файлов

Новая команда с именем `mv` (MoVe) перемещает файлы вместо их копирования. Синтаксис команды очевиден.

```
/home/larry/foo# mv termcap sells
/home/larry/foo# ls -F
bells    sells    shells
/home/larry/foo#
```

Обратите внимание, что теперь termcap уже не существует, а на его месте файл `sells`. Это можно использовать для переименования файлов, что мы сейчас и сделали. Но можно и переносить файлы в совсем другие каталоги.

Внимание! Команды `mv` и `cp` уничтожат содержимое файла в который они пишут (если он существовал), не спрашивая вашего разрешения. Будьте внимательны, когда вы переносите файл в другой каталог: там уже может существовать файл с таким именем и вы его затрете.

3.3.6 Удаление файлов и каталогов

Мы тут с вами ”нарифмовали” ненужных файлов, изучая работу команды `ls`. Для удаления файлов используется команда `rm` (ReMove).

---

[18] (прим. переводчика: несколько файлов одной командой cp можно скопировать только в каталог; в файл можно скопировать только один файл).

```
/home/larry/foo# rm bells sells
/home/larry/foo# ls -F
shells
/home/larry/foo#
```

У нас ничего не осталось, кроме `shells`, но не будем переживать. Обратите внимание, что команда `rm` не будет вас переспрашивать перед удалением, так что будьте осторожны.

Родственная `rm` команда `rmdir`. Эта команда удаляет каталоги, но только пустые каталоги. Если в каталоге есть хоть какие-нибудь файлы или подкаталоги, она распишется в бессилии.

3.3.7 Рассматривание файлов

Команды `more` и `cat` используются для просмотра содержимого файлов. `more` выдает файл на дисплей "поэкранно", в то время, как cat выдает весь файл разом.[19]

Чтобы посмотреть файл `shells`, используем команду

```
/home/larry/foo# more shells
```

При использовании команды `more` нажимайте клавишу пробел для перехода к следующей странице и *b* для возврата к предыдущей. Нажав *q*, вы выйдете из more.

А теперь попробуйте команду `cat` etc/termcap/. Текст промелькнет слишком быстро, чтобы успеть его прочитать. На самом деле команда "`cat`" (conCATenate) в основном используется для других целей, для той же конкатенации нескольких файлов. Это в дальнейшем будет обсуждаться.

3.3.8 Получение оперативной помощи

Практически каждый UNIX имеет то, что называется "Руководство" - `man` ("manual pages"). Эта команда `man` содержит документацию на различные команды системы, ресурсы, конфигурационные файлы. Например, если вы хотите найти информацию о других опциях команды `ls`, введите

```
/home/larry# man ls
```

---

[19] (прим. переводчика: если файл многострочный, то, при использовании команды cat файл промелькнет и на экране останутся последние строки).

и вам на экран будут выведены страницы Руководства.

К сожалению, большинство страниц руководства написаны с ориентацией на пользователей, имеющих некоторые представления о работе соответствующих команд. Поэтому страницы Руководства обычно содержат справочные данные по командам, а не учебный материал.

Но Руководство неоценимо для освежения памяти, если вы забыли синтаксис команды. Руководство может также много рассказать вам о командах, которые мы даже не упомянем в этой книге.

Я предлагаю вам посмотреть в Руководстве те команды, которые мы уже обсуждали и все, с которыми мы будем встречаться. Вы обнаружите, что не на все команды есть Руководство. Тому несколько причин. Одна - некоторые страницы Руководства еще просто не написаны (*the Linux Documentation Project* - программа подготовки документации для Linux как бы отвечает за решение этой проблемы. Мы уже собрали большую часть документации). Во-вторых, команда может быть внутренней командой shell или синонимом (alias), что обсуждалось в Разделе 3.2.4, в каждом из этих случаев для них нет собственных страниц. Возьмем для примера `cd`, которая является внутренней командой shell. Shell выполняет эту команду, но она не имеет своей отдельной программы.

3.4 Краткая информация о базовых командах

Этот раздел представляет некоторые наиболее полезные базовые команды UNIX, включая те, о которых говорили в предыдущем разделе.

Обратите внимание, что опции обычно начинаются с “-” и во многих случаях несколько однобуквенных опций могут следовать за одним минусом, записанные слитно. Например, вместо использования `ls -l -F`, можно использовать `ls -lF`.

Вместо перечисления всех возможных опций каждой команды, мы будем говорить только о тех, которые полезны или важны в данное время. Действительно, большинство из этих команд имеет большое число опций (большинство из которых никогда не используется). Вы можете для каждой команды с помощью man посмотреть все возможные опции.

Обратите также внимание на то, что многие из команд берут список файлов или каталогов, как аргументы, обозначенные как “<file1> ... <fileN>”. Например, команда `cp` берет в качестве аргументов список файлов, которые надо копировать, за которыми следует имя целевого файла или каталога. При копировании нескольких файлов в качестве целевого может выступать только каталог.

cd

Изменяет текущий рабочий каталог.

Синтаксис: `cd <directory>`;

`<directory>` - каталог, в который перейти (“.” ссылается на текуший каталог, “..” - на родительский каталог).

Пример: `cd ../foo` переводит из текущего каталога в `../foo`.

ls

Выдает информацию о файлах в каталоге.

Синтаксис: `ls <file1> ... <fileN>`

Где `<file1> ... <fileN>` имена файлов или каталогов, информацию про которые надо выдать.

Опции: Здесь больше опций, чем вы думаете. Наиболее часто используемые: `-F` (для представления информации о типах файлов), и `-l` (выдает в длинном (“long”) формате информацию о размерах файлов, владельцах, правах доступа и т.д. В деталях это будет обсуждаться далее).

Пример: `ls -lF /home/larry` выдаст содержимое каталога `/home/larry`.

cp

Копирует файл(ы) в файл или каталог.

Синтаксис: `cp <file1> ... <fileN> <destination>`

Где `<file1> ... <fileN>` имена копируемых файлов, а `<destination>` файл или каталог, в который копируют.

Пример: `cp ../frog joe` копирует файл `../frog` в файл или каталог `joe`.

mv

Перемещает файл(ы) в другой файл или каталог. Эта команда не эквивалентна копированию с последующим уничтожением оригинала. Она может быть использована для переименования файлов, как команда `RENAME` из MS-DOS.

Синтаксис: `mv <file1> ... <fileN> <destination>`

Где `<file1> ... <fileN>` имена перемещаемых файлов, а `<destination>` имя файла или каталога, в который перемещают.

Пример: `mv ../frog joe` перемещает файл `../frog` в файл или каталог `joe`.

rm

Удаляет файлы. Имейте в виду, когда в UNIX удаляются файлы, они невосстановимы (не как в MS-DOS, где вы можете ”разудалить” файл).

Синтаксис: `rm <file1> ... <fileN>`

Где `<file1> ... <fileN>` имена удаляемых файлов.

Опции: `-i` потребует вашего подтверждения перед удалением файла.

Пример: `rm -i /home/larry/joe /home/larry/frog` удаляет файлы `joe` и `frog` в каталоге `/home/larry`.

mkdir

Создает новые каталоги.

Синтаксис: `mkdir <dir1> ... <dirN>`

Где `<dir1> ... <dirN>` создаваемые каталоги.

Пример: `mkdir /home/larry/test` создает каталог `test` в каталоге `/home/larry`.

rmdir

Эта команда удаляет пустые каталоги. При использовании `rmdir` ваш текущий рабочий каталог должен находиться вне удаляемого каталога.

Синтаксис: `rmdir <dir1> ... <dirN>`

Где `<dir1> ... <dirN>` удаляемые каталоги.

Пример: `rmdir /home/larry/papers` удаляет каталог `/home/larry/papers`, если он пустой.

man

Выдает страницу Руководства по данной команде или ресурсу. (здесь "ресурс" - это любая системная утилита, которая не является командой, например библиотечная функция).

Синтаксис: `man <command>`

Где `<command>` имя команды или ресурса, о котором запрашивается информация.

Пример: `man ls` - дает помощь по команде `ls`.

more

Выдает содержимое названных файлов поэкранно.

Синтаксис: `more <file1> ... <fileN>`

Где `<file1> ... <fileN>` отображаемые файлы.

Пример: `more papers/history-final` представляет файл `papers/history-final`.

cat

Используется для конкатенации файлов. `cat` используется также для выдачи полного содержания файла разом

Синтаксис: `cat <file1> ... <fileN>`

Где `<file1> ... <fileN>` выдаваемые файлы.

Пример: `cat letters/from-mdw` выдает на дисплей файл `letters/from-mdw`.

echo

Просто повторяет аргументы.

Синтаксис: `echo <arg1> ... <argN>`

Где `<arg1> ... <argN>` ”повторяемые” аргументы.

Пример: `echo ”Hello world”` выдает на экран “`Hello world`”.

grep

выдает все строки в названном файле(лах), которые содержат заданный образец.

Синтаксис: `grep <pattern> <file1> ... <fileN>`

Где `<pattern>` - образец (представленный регулярным выражением) и `<file1> ... <fileN>` - файлы, в которых производится поиск.

Пример: `grep loomer /etc/hosts` выдаст все строки, в которых файл `/etc/hosts`, содержит образец “`loomer`”.

3.5 Исследование файловой системы

Файловая система есть собрание файлов и иерархия каталогов. Я обещал поводить вас по файловой системе - и время настало. У вас достаточно интеллекта и знаний извлечь пользу из того, что я говорю и у вас есть карта дорог. (Смотрите Рис. 3.2.8).

Перво-наперво вернемся в корневой каталог (`cd /`) и сделаем `ls -F`. Вы, очевидно, увидите каталоги: `bin`, `dev`, `etc`, `home`, `install`, `lib`, `mnt`, `proc`, `root`, `tmp`, `user`, `usr` и `var`. (Можете увидеть и несколько отличный вариант - не волнуйтесь, различные версии Linux могут иметь отличия).

Присмотримся к каждому каталогу.

/bin

`bin` - это сокращенно от “binaries” (т.е. двоичные или выполняемые файлы). Здесь находится много важных системных программ. Используйте команду “`ls -F/bin`” чтобы посмотреть имеющийся здесь список файлов. Вы можете обнаружить здесь уже знакомые вам команды, вроде `cp`, `ls` и `mv`. Это и есть программы соответствующих команд. Когда, например, вы используете команду `cp`, вы выполняете программу `/bin/cp`.

Используя `ls -F`, вы увидите, что большинство (если не все) файлов в `/bin` имеют справа от имени звездочку (“`*`”). Это говорит о том, что файлы выполняемые, как описано в Разделе 3.3.2.

/dev

Следующая остановка на нашем пути - `dev`. Вновь посмотрите на содержимое с помощью `ls -F`.

"Файлы" в `/dev` известны как **драйверы устройств** - они используются для доступа к устройствам и ресурсам системы, таким как диски, модемы, память и т.д. Например, как вы можете читать данные из файла, точно также вы можете читать входные сигналы от мыши, имея доступ к `/dev/mouse`. Имена файлов, начинающиеся на `fd` - это дисководы гибких дисков. `fd0` - первый дисковод, `fd1` - второй. Теперь самые шустрые из вас заметят, что здесь имеется больше дисководов, чем те два, которые мною упоминались: они представляют специфические типы дисководов. Например, `fd1H1440` представляет доступ к high-density, 3.5" дискетам на дисководе 1.

Вот перечень некоторых из наиболее используемых файлов устройств.

- `/dev/console/` относится к системной консоли, т.е. к монитору, напрямую связанному с системой.
- Различные `/dev/ttyS` и `/dev/cua` устройства используются для доступа к последовательным портам. Например, `/dev/ttyS0` относится к "COM1" под MS-DOS. Устройства `/dev/cua` относятся к "звонящим" ("callout") устройствам, которые используются совместно с модемами.
- Устройства, имена которых начинаются с `hd`, имеют доступ к жестким дискам. `/dev/hda` относится ко *всему* первому жесткому диску, а `hda1` только к *первому разделу* `/dev/hda`.
- Устройства с именами, начинающимися на `sd` - SCSI-драйверы. Если у вас SCSI жесткий диск, вместо доступа к нему через `/dev/hda`, вы будете обращаться к `/dev/sda`. SCSI ленты доступны через устройства `st`, а SCSI CD-ROM через `sr`.
- Устройства `lp` обеспечивают доступ к параллельным портам. `/dev/lp0` относится к "LPT1" в MS-DOS.
- `/dev/null` используется как "черная дыра" - любые данные, посланные сюда, канут в Лету. Если вы хотите подавить вывод команды на экран, вы можете перенаправить этот вывод в `/dev/null`. Мы об этом позже еще поговорим.
- Устройства с именами `/dev/tty` относятся к "виртуальным консолям" вашей системы (доступ путем нажатия $alt - F1$, $alt - F2$ и т.д.). `/dev/tty1` соответствует первой VC, `/dev/tty2` соответствует второй и т.д.
- Устройства, чьи имена начинаются на `/dev/pty`, это "псевдотерминалы". Они используются для входа с удаленных "терминалов". Например, если ваша машина в сети, вход к вам по telnet будет использовать одно из устройств `/dev/pty`.

/etc

`/etc` содержит множество всевозможных системных файлов конфигурации. Они включают `/etc/passwd` (файл паролей), `/etc/rc` (командный файл инициализации) и т.д.

/sbin

`/sbin` используется для хранения важных системных двоичных файлов, используемых системным администратором.

/home

`home` содержит домашние каталоги пользователей. Например, `/home/larry` - домашний каталог пользователя “`larry`”. На вновь инсталлированной системе этот каталог может быть пуст в связи с временным отсутствием зарегистрированных пользователей.

/lib

`/lib` содержит образы **разделяемых библиотек** (**shared library images**). Эти файлы содержат код, который могут использовать многие программы. Вместо того, чтобы каждая программа имела свою собственную копию этих выполняемых файлов, они хранятся в одном общедоступном месте - в `/lib`. Это позволяет сделать выполняемые файлы меньше и сэкономит место в системе.

/proc

`/proc` - это ”виртуальная файловая система”, в которой файлы хранятся в памяти, а не на диске. Они связаны с различными **процессами**, происходящими в системе, и позволяют получить информацию о том, что делают программы и процессы в указанное время. Более детально мы рассмотрим это в Разделе 3.11.1.

/tmp

Многие программы нуждаются в создании рабочих файлов, которые нужны короткое время. Каноническое место для этих файлов в `/tmp` (там обычно чаще проводится уборка мусора).

/usr

`/usr` - это очень важный каталог. Он состоит из ряда подкаталогов, которые в свою очередь содержат наиболее важные и полезные программы и файлы конфигурации, используемые системой.

Различные каталоги, описанные выше, необходимы для нормального функционирования системы, но большинство вещей, содержащихся в `/usr` необязательны для системы. Но это такие необязательные вещи, которые делают систему полезной и интересной. Без `/usr` вы бы имели достаточно занудную систему, содержащую только программы, вроде `cp` и `ls`. `/usr` содержит много больших программных пакетов и конфигурационных файлов, которые их сопровождают.

/usr/X386

`/usr/X386` содержит The X Window System, если вы ее инсталлировали. The X Window System - это мощная графическая среда, которая содержит большое количество графических утилит и программ, отображающих "окна" на вашем экране. Если вы знакомы с Microsoft Windows или Macintosh environments, то X Windows будет выглядеть весьма похоже. Каталог `/usr/X386` содержит все выполняемые и конфигурационные файлы X Window, а также файлы поддержки. Более детально это будет обсуждаться в Разделе 5.1.

/usr/bin

`/usr/bin` настоящее хранилище для различных программ UNIX. Он содержит большинство выполняемых программ, которых нет ни в каких других местах, например, в том же `/bin` их нет.

/usr/etc

Точно также, как и `/etc`, содержит всевозможные системные программы и конфигурационные файлы. `/usr/etc` содержит даже больше утилит и файлов. В общем, файлы, находящиеся в `/usr/etc` несущественны для системы, в отличие от тех, которые находятся в `/etc`, и очень существенны.

/usr/include

/usr/include содержит **include-файлы** для компилятора Си. Эти файлы (большинство имен которых заканчивается на .h (от слова "header") об'являют имена структур данных, подпрограмм и констант, используемых при написании программ на Си. Те файлы, которые находятся в `/usr/include/sys` в общем случае используются при программировании на системном уровне UNIX. Если вы знакомы с языком программирования Си, здесь вы найдете такие хэдеры (фрагменты программ, вставляемые обычно в начало программы), `stdio.h`, которые описывают такие функции, как `printf()`.

/usr/g++-include

`/usr/g++-include` содержит include-файлы для компилятора Си++ (очень похожие на `/usr/include`).

/usr/lib

`/usr/lib` содержит библиотеки-"заглушки" и "статические" библиотеки, эквивалентные файлам из `/lib`. При компиляции программа "связывается" с библиотеками, находящимися в `/usr/lib`, которые в свою очередь направляют программы обращаться в `/lib`, если им нужет актуальный код. Кроме того, многие другие программы хранят в `/usr/lib` свои конфигурационные файлы.

/usr/local

`/usr/local` в большой степени похож на `/usr` - он содержит различные программы и файлы, несущественные для системы, но превращающие ее в удовольствие и восторг. В общем, эти программы, находящиеся в `/usr/local` специализируются на специфике вашей системы, т.е. `/usr/local` сильно отличается в различных UNIX. Здесь вы найдете такие большие программные пакеты, как TeX (система форматирования документов) и Emacs (большой и мощный редактор), если вы их установите.

/usr/man

Этот каталог содержит страницы Руководства. Здесь два подкаталога для каждого "раздела" Руководства. (С помощью команды "man man" вы можете получить более подробную информацию). Например, `/usr/man/man1` содержит исходные тексты (неотформатированный оригинал) страниц Руководства в разделе 1 и `/usr/man/cat1` содержит отформатированные страницы для раздела 1.

/usr/src

`/usr/src` содержит исходные коды (неоткомпилированные программы) для различных программ вашей системы. Наиболее важная вещь здесь, это `/usr/src/linux`, содержащий исходные коды ядра Linux.

/var

`/var` содержит каталоги, которые часто меняются в размере или имеют тенденцию быстро расти. Многие из этих каталогов "квартировались" в `/usr`, но поскольку мы стремимся сделать его достаточно стабильным, каталоги, которые часто меняются были перенесены в `/var`. К числу таких каталогов относятся:

/var/adm

`/var/adm` содержит различные файлы, интересные системному администратору, специфические системные файлы, фиксирующие ошибки и проблемы, возникающие в системе. Другие файлы фиксируют входы в систему, как и неудачные попытки войти. Это будет обсуждаться в Главе 4.

/var/spool

`/var/spool` содержит файлы, которые предварительно формируются для других программ. Например, если ваша машина подключена к сети, входная почта будет помещаться в `/var/spool/mail` до тех пор, пока вы не прочитаете ее или не удалите. Входящие и исходящие новости помещаются в `/var/spool/news` и т.д.

3.6 Типы оболочек

Как я уже много раз говорил, UNIX - это многозадачная, многопользовательская операционная система. Многозадачность *очень* полезна - однажды привыкнув к ней, вы будете всегда ее использовать. Прежде всего, вы сможете выполнять задачи в фоновом режиме, переключать задачи и об'единять их в конвейер, достигая сложных результатов простыми средствами.

Многие из возможностей, которые мы будем обсуждать в этом разделе, обеспечиваются самой оболочкой (shell). Будьте внимательны, не путайте UNIX (фактическую операционную систему) с оболочкой - оболочка, это лишь интерфейс с находящейся за ней системой. Оболочка обеспечивает выполнение громадного числа функций помимо собственно UNIX.

Оболочка - это не только интерпретатор интерактивных команд, которые вы можете ввести, получив от оболочки подсказку (готовности принимать команды). Это также мощный командный язык, который позволяет писать программы (**shell-scripts**), об'единяющие несколько команд в **командный файл**. Пользователи MS-DOS почувствуют здесь нечто схожее с ‘‘batch-файлами’’. Использование программ на языке оболочки (shell) - это очень мощное средство, которое позволяет автоматизировать и и существенно повысить эффективность использования UNIX. Смотрите дополнительно в Разделе 3.13.1.

Существует несколько типов оболочек в мире UNIX. Две главные - это ‘‘Bourne shell’’(shell Баурна) и ‘‘C shell’. Shell Баурна (или просто shell) использует командный синтаксис, похожий на первоначально для UNIX придуманный (вроде UNIX System III). В большинстве UNIX-систем shell Баурна имеет имя `/bin/sh` (где `sh` сокращение от ‘‘shell’’). C shell использует иной синтаксис, чем-то напоминающий синтаксис языка программирования Си. В большинстве UNIX-систем он имеет имя `/bin/csh`.

В Linux есть несколько вариаций этих оболочек. Две наиболее часто используемые, это Новый Shell Баурна (Bourne Again Shell) или ‘‘Bash’’ (`/bin/bash`) и Tcsh (`/bin/tcsh`). Bash - это развитие прежнего shell с добавлением многих полезных возможностей, частично содержащихся в C shell. Поскольку Bash можно рассматривать как надмножество синтаксиса прежнего shell, любая программа, написанная на добром старом shell Баурна должна работать и в Bash. Для тех, кто предпочитает использовать синтаксис C shell, Linux поддерживает Tcsh, который является расширенной версией C shell.

Тип оболочки, которую вы решили использовать - это почти как выбор религии. Некоторые предпочитают синтаксис shell Баурна с дополнительными возможностями, предоставляемыми Bash, а некоторые - более структурированный синтаксис C shell. Для "нормальных" команд, таких как `cp` и `ls`, тип используемого вами shell никакой роли не играет. Только когда вы начнете писать командные файлы или использовть некоторые новые свойства оболочек, различия между ними становятся существенными.

При обсуждении далее некоторых свойств оболочек мы будем обращать внимание на различие между Баурновским shell и C shell. (Если вам это действительно очень интересно, почитайте Руководство по поводу `bash` и `tcsh`).

3.7 "Уайлдкард" - "дикая карта"

Ключевое свойство большинства оболочек Unix - это способность ссылаться сразу более, чем на один файл, используя для этого специальные символы. Эти, так называемые "дикие карты" (**wildcards**), позволяют ссылаться, скажем, на все файлы, содержащие символ "`n`".[20]

Уайлдкард "`*`" относится к любому символу или строке символов в имени файла. Например, когда вы используете символ "`*`" в имени файла shell заменяет ее всеми возможными именами файлов из каталога, на который вы ссылаетесь. Вот простенький пример. Предположим, что Larry имеет файлы `frog`, `joe` и `stuff` в своем текущем каталоге:

```
/home/larry# ls
frog     joe     stuff
/home/larry#
```

Для обращения сразу ко всем файлам с буквой "`o`" в имени, мы можем использовать команду

```
/home/larry# ls *o*
frog     joe
```

---

[20] (прим. переводчика: Мне не известен хороший перевод этой идиомы (wildcards), наиболее часто у нас встречается "генераторы" и "расширители" символов - но это тяжеловато. Чтобы далее не испытывать мучений - буду использовать слово "уайлдкард". Кстати, и оболочку удобнее далее именовать как shell, так легче воспринимается то, что это язык программирования).

```
/home/larry#
```

Как видите, ‘‘*’’ уайлдкард была заменена всеми возможными именами файлов из имевшихся в текущем каталоге.

Использование просто ‘‘*’’даст совпадение со всеми именами, поскольку все символы совпадают с уайлдкард.

```
/home/larry# ls *
frog     joe     stuff
/home/larry#
```

Вот еще несколько примеров.

```
/home/larry# ls f*
frog
/home/larry# ls *ff
stuff
/home/larry# ls *f*
frog     stuff
/home/larry# ls s*f
stuff
/home/larry#
```

Процесс замены ‘‘*’’ на имена файлов называется расширением уайлдкард и выполняется shell. Это важно: конкретные команды, вроде `ls`, никогда не видят ‘‘*’’ в своем списке параметров. Shell, расширяя уайлдкард, включает в список параметров все имена, прошедшие сравнение с шаблоном. Так что команда

```
/home/larry# ls *o*

 расширяется shell до фактической

/home/larry# ls frog joe
```

Одно важное замечание относительно ‘‘*’’ уайлдкард. Использование этой уайлдкард не даст совпадения с именами файлов, которые начинаются с точки (‘‘.’’). Эти файлы воспринимаются как "спрятанные", хотя на самом деле их никуда не прятали. Они не показываются в списке, выдаваемом нормальной командой `ls` и не выбираются при использовании ‘‘*’’ уайлдкард.

Вот пример. Мы уже упоминали, что каждый каталог имеет два специальных файла: ‘‘.’’ - указание на текущий каталог и ‘‘..’’ - указание на родительский каталог. Однако, если вы используете команду `ls`, эти два файла не будут отображены.

```
/home/larry# ls
frog     joe     stuff
/home/larry#
```

Если вы используете опцию `-a` в команде `ls`, то вы сможете отобразить имена, начинающиеся на “`.`”:

```
/home/larry# ls -a
.      ..      .bashprofile      .bashrc      frog        joe
stuff
/home/larry#
```

Как видим, два специальных файла “`.`” и “`..`”, также, как два других ”спрятанных” файла - `.bash_profile` и `.bashrc`. Эти два файла используются при входе larry в систему. Более подробно о них в Разделе 3.13.3.

Обратите внимание, что когда мы используем “`*`” уайлдкард, ни один из файлов, с именами, начинающимися на “`.`” не отображается.

```
/home/larry# ls *
frog     joe     stuff
/home/larry#
```

Это мера предосторожности: если “`*`” уайлдкард выбирала бы имена файлов, начинающиеся на “`.`”, она бы также выбрала имена “`.`” и “`..`”. Но это может быть опасно при выполнении ряда команд.

Другой уайлдкард является “`?`”. “`?`” уайлдкард позволяет подставить строго один символ. Так “`ls ?`” выдаст на только имена файлов, состоящие из одного символа, а “`ls termca?`” выдаст “`termcap`”, но не выдаст на экран “`termcap.backup`”. Вот еще один пример:

```
/home/larry# ls j?e
joe
/home/larry# ls f??g
frog
/home/larry# ls ????f
stuff
/home/larry#
```

Как видите, уайлдкард позволяет описывать много файлов за один раз. При обзоре простейших команд в Разделе 3.4 мы говорили, что команды `cp` и `mv` могут копировать или перемещать множества файлов за один раз. Например,

```
/home/larry# cp /etc/s* /home/larry
```

скопирует все файлы в `/etc` , начиная с “s”в каталог `/home/larry`. Формат команды `cp` на самом деле

```
cp file1 ... fileN destination
```

где <`file1`> `...` <`fileN`> - список копируемых файлов, а <`destination`> это файл или каталог, в который производится копирование. mv имеет аналогичный синтаксис.

Обратите внимание, что если производится копирование или перемещение более, чем одного файла, <`destination`> должен быть каталогом. В файл скопировать или переместить можно только один файл.

3.8 Трубопроводы UNIX

3.8.1 Стандартный вход и стандартный выход

Многие команды UNIX получают информацию с так называемого **стандартного входа** и посылают информацию на (опять же) так называемый **стандартный выход**. (Для них часто используются сокращения “**stdin**” и “**stdout**” соответственно). Ваш shell организует дело так, что стандартным входом служит клавиатура, а стандартным выходом - экран.

Вот пример использования команды `cat`. Нормально cat читает данные из файлов, чьи имена даны в командной строке и посылает эти данные прямехонько на stdout. Поэтому при выполнении команды

```
/home/larry/papers# cat history-final masters-thesis
```

на экран пойдет файл `history-final`, а за ним следом `masters-thesis`.

Но если команде cat не даны имена файлов в качестве параметров, она читает данные с stdin и опять же посылает на stdout. Вот пример.

```
/home/larry/papers# cat
Hello there.
Hello there.
Bye.
Bye.
[ctrl-D]
/home/larry/papers#
```

Как видите, каждая строка, которую напечатал пользователь, немедленно выдается командой cat на экран. При вводе со стандартного входа команда знает, что ввод закончен тогда, когда она получит в каком-то виде сигнал EOT (End-Of-Text). Обычно он обеспечивается нажатием *ctrl* – *D*.

Вот другой пример. Команда сортировки sort читает построчно текст (здесь опять с stdin, поскольку имена файлов в параметрах не указаны, и посылает отсортированный результат на stdout. Попробуйте так.

```
/home/larry/papers# sort
bananas
carrots
apples
[ctrl-D]
apples
bananas
carrots
/home/larry/papers#
```

Теперь мы можем упорядочить наш список продуктов, подлежащих закупке, в лексикографическом порядке... ну разве UNIX не полезная вещь?

3.8.2 Перенаправление входа и выхода

Теперь, предположим, что мы хотим послать результат сортировки в файл, чтобы где-то сохранить список планируемых покупок. Shell дает нам возможность **перенаправлять** стандартный выход в файл, используя символ “>”. Вот как это работает.

```
/home/larry/papers# sort  shopping-list
bananas
carrots
apples
[ctrl-D]
/home/larry/papers#
```

Как вы можете видеть, результат работы команды `sort` не отображается на экране, вместо этого он сохраняется в файле `shopping-list` (список покупок). Давайте посмотрим на этот файл.

```
/home/larry/papers# cat shopping-list
apples
bananas
```

```
carrots
/home/larry/papers#
```

Теперь мы можем не только сортировать (упорядочивать) список планируемых покупок, но и сохранять его! Но предположим, что мы хранили наш неотсортированный исходный закупочный список в файле под именем `items`. Один из способов сортировки и сохранения его, это отсортировать файл с данным именем, вместо получения файла со стандартного входа, и перенаправить стандартный выход в файл. Например так

```
/home/larry/papers# sort items  shopping-list
/home/larry/papers# cat shopping-list
apples
bananas
carrots
/home/larry/papers#
```

Но это можно сделать и по-другому. Перенаправлен может быть не только стандартный выход, но также и стандартный *вход*, используя символ “<”.

```
/home/larry/papers# sort  items
apples
bananas
carrots
/home/larry/papers#
```

Технически, `sort < items` эквивалентно `sort items`, но последний вариант позволяет нам продемонстрировать сказанное: `sort < items` ведет себя так, словно данные файла `items` были напечатаны на клавиатуре. shell обслуживает перенаправление. `sort` не было дано имя файла (`items`) и команда читала со стандартного входа, как будто шел ввод с клавиатуры.

Это иллюстрирует концепцию **фильтра**. Фильтр, это программа, которая получает данные со стандартного входа, обрабатывает их каким-то образом и посылает результат обработки на стандартный выход. С помощью перенаправления стандартные вход и выход могут быть переведены на файлы. `sort` - простейший фильтр: она сортирует входные данные и посылает результат на стандартный выход. `cat` - даже еще проще: она ничего не делает со входными данными, а только выдает все, что не поступит, на выход.

3.8.3 Использование конвейера

Мы уже показали, как использовать команду sort в качестве фильтра. Но эти примеры предполагали, что вы откуда-то получили данные в файл, или ввели данные с клавиатуры своими собственными руками. А что, если данные, которые вы хотите отсортировать, являются выходными данными другой программы, например, такой как `ls`? Если вы используете при сортировке опцию `-r`, данные будут расположены в порядке, обратном лексикографическому. Если вы хотите получить перечень файлов вашего каталога в обратном порядке, один из способов сделать это будет:

```
/home/larry/papers# ls
english-list
history-final
masters-thesis
notes
/home/larry/papers# ls  file-list
/home/larry/papers# sort -r file-list
notes
masters-thesis
history-final
english-list
/home/larry/papers#
```

Здесь мы сохранили результат работы команды `ls` в файле, а затем выполнили `sort -r` над этим файлом. Но это очень коряво выглядит и требует создания временного файла для хранения результата работы `ls`.

Выход из положения дает трубопровод (**pipeline**) (прим. переводчика: в нашей литературе принят термин ”**конвейер**”, так далее и будем переводить ”pipeline”). Конвейер - это еще одно замечательное свойство shell, которое позволяет связывать последовательность команд в конвейер, где stdout первой команды посылается прямо на stdin второй команды и так далее. Здесь мы хотим послать stdout команды `ls` на stdin команды sort. Символ “`|`” олицетворяет конвейер:

```
/home/larry/papers# ls | sort -r
notes
masters-thesis
history-final
english-list
/home/larry/papers#
```

Эта команда намного короче и, очевидно, проще набирается. Другой полезный пример. Команда

```
/home/larry/papers# ls /usr/bin
```

выдает на дисплей длинный список имен файлов, большинство из которых слишком быстро промелькнет на экране, чтобы вы успели прочитать их. Давайте подключим к просмотру перечня имен файлов каталога /usr/bin команду more.

```
/home/larry/papers# ls /usr/bin | more
```

Теперь вы можете постранично листать файл в свое удовольствие.

Но чудеса на этом не кончаются! Мы можем связать в конвейер более, чем две команды. Команда head представляет из себя фильтр, который отображает первые строки входного потока (здесь, пришедшего по конвейеру). Если мы хотим отобразить последнее имя текущего каталога, упорядоченного по алфавиту, мы можем написать:

```
/home/larry/papers# ls | sort -r | head -1
notes
/home/larry/papers#
```

где `head -1` просто выдает первую строку получаемого входного потока (в данном случае это отсортированный в обратном порядке перечень имен файлов текущего каталога, выданных командой `ls`).

3.8.4 Перенаправление с добавлением

Использование “>” для перенаправления выхода смертельно для файла, в который происходит перенаправление (если было, что уничтожать), другими словами

```
/home/larry/papers# ls  file-list
```

уничтожает прежнее содержимое файла `file-list`. Если вместо этого использовать символ перенаправления “>>”, выход будет добавлен к содержимому названного файла (вместо того, чтобы быть записаным на место старого).

```
/home/larry/papers# ls  file-list
```

добавит выходную информацию команды ls в файл `file-list`.

Имейте в виду, что перенаправления и конвейер, это средства, предоставляемые оболочкой shell, это синтаксис shell и символы “>”, “>>” и “|” не имеют никакого отношения к командам, как таковым.

3.9 Права доступа к файлам

3.9.1 Концепция прав доступа

Поскольку UNIX - многопользовательская система, чтобы защитить файлы каждого пользователя от дурного влияния других пользователей, UNIX поддерживает механизм, известный, как **система прав доступа к файлам**. Этот механизм позволяет каждому файлу приписать конкретного владельца. Как пример, поскольку Larry создал файлы в своем домашнем каталоге, именно Larry владелец этих файлов и имеет к ним доступ.

UNIX позволяет также совместно использовать файлы нескольким пользователям и группам пользователей. Если Larry так пожелает, он может закрыть доступ к своим файлам так, что никто другой не сможет к ним подступиться. Однако в большинстве систем по умолчанию другим пользователям разрешается читать ваши файлы, но запрещается изменять или удалять.

Как об'яснялось выше, каждый файл имеет конкретного владельца. Но, кроме того файлами, также владеют конкретные **группы** пользователей, которые определяются при регистрации пользователей в системе. Каждый пользователь становится членом как минимум одной группы пользователей. Системный администратор может даровать пользователю доступ более, чем к одной группе.

Группы обычно определяются типами пользователей данной машины. Например, в университетском UNIX пользователи могут быть разбиты на группы `студент`, `преподаватель`, `руководство`, `гость`.[21]

Есть также несколько системно-зависимых групп (вроде `bin` и `admin`), которые используются самой системой для управления доступом к ресурсам. Очень редко обычный пользователь принадлежит к этим группам.

Права доступа подразделяются на три типа: *чтение* (*read*), *запись* (*write*) и *выполнение* (*execute*). Эти типы прав доступа могут быть предоставлены трем классам пользователей: владельцу файла, группе, в которую входит владелец, и всем (прочим) пользователям.

Разрешение на чтение позволяет пользователю читать содержимое файлов, а в случае каталогов - просматривать перечень имен файлов в каталоге (используя, например, `ls`). Разрешение на запись позволяет пользователю писать в файл и изменять его. Для каталогов это дает право создавать в каталоге новые файлы и каталоги, или удалять файлы в этом каталоге. Наконец, разрешение на выполнение позволяет пользователю выполнять файлы (как бинарные программы, так и командные файлы). Разрешение на выполнение применительно к каталогам означает возможность выполнять команды вроде cd.

[21] (прим. переводчика: осмелюсь предположить, что в отечественной книге перечисление примеров групп было бы начато с группы "руководство"...).

3.9.2 Интерпретация прав доступа

Давайте рассмотрим пример, демонстрирующий работу с правами доступа. Используя команду `ls` с опцией `-l` можно получить на экране перечень файлов данного каталога в "длинном" формате, включающем информацию о правах доступа.

```
/home/larry/foo# ls -l stuff

-rw-r--r--   1 larry    users         505 Mar 13 19:05 stuff

/home/larry/foo#
```

Первое поле в выведенной строке представляет права доступа. Третье поле - владельца файла (`larry`) и четвертое - группу (`users`). Очевидно, что последнее поле есть имя файла (`stuff`), а остальные поля мы обсудим позже.

Этим файлом владеет `larry`, и он принадлежит группе `users`. Давайте посмотрим на права доступа. В строке `-rw-r--r--` по порядку указаны права владельца, группы и всех прочих.

Первыый символ этой строки прав доступа (“`-`”) представляет тип файла. Символ “`-`” означает, что это обычный файл (в противоположность каталогу или специальному файлу какого-то устройства). Следующие три позиции (“`rw-`”) представляют права доступа, которые имеет владелец файла `larry`. Символ “`r`”означает “read”(читать), “`w`” - “write”(писать). Таким образом `larry` может читать файл `stuff` и писать в него.

Как мы уже упоминали, кроме разрешений на чтение и запись существует разрешение на выполнение “execute” - представляемое символом “`x`”. Но в данном случае на этой позиции “`-`”, так что у Larry нет прав на выполнение этого файла. И это чудесно, файл `stuff` совсем даже не является программой. Разумеется, поскольку Larry владеет файлом, он может дать сам себе разрешение на выполнение этого файла, если захочет. Мы эту процедуру скоро обсудим.

Следующие три символа `r--` представляют права доступа группы для этого файла. Эта группа имеет имя `users`. Поскольку тут есть только “`r`”, любой пользователь этой группы может только читать файл.

Последние три символа представляют ту же комбинацию `r--`, то есть для всех прочих доступно чтение этого файла и запрещены запись и выполнение.

Вот еще несколько примеров на права доступа.

-rwxr-xr-x

Владелец файла может читать, писать и выполнять файл. Члены группы и все прочие пользователи могут читать и выполнять файл.

-rw-------

Владелец файла может читать и писать в файл. Всем остальным доступ к файлу закрыт.

-rwxrwxrwx

Все могут читать писать и выполнять файл.

3.9.3 Зависимости

Важно заметить, что права доступа, которые имеет файл зависят также от прав доступа к каталогу, в котором этот файл находится. Например, даже если файл имеет `-rwxrwxrwx`, другие пользователи не смогут до него добраться, если у них не будет прав на чтение и выполнение каталога, в котором находится файл. Например, если Larry захочет ограничить доступ ко всем своим файлам, он может может просто изменить права доступа своего домашнего каталога `/home/larry` на `drwx------`. Таким образом, никто другой не будет иметь доступ в его каталог, а следовательно посторонним будут недоступны и все файлы. Так что Larry может не заботиться об индивидуальной защите своих файлов.

Другими словами, чтобы иметь доступ к файлу, вы должны иметь доступ ко всем каталогам, лежащим на пути от корня к этому файлу, а также разрешение на доступ собственно к этому файлу.

Обычно пользователи UNIX весьма открыты всеми своими файлами. Обычно файлам устанавливается защита `-rw-r--r--`, которая позволяет другим пользователям читать файлы, но ни коим образом их не менять. Каталогамм обычно устанавливаются права доступа `drwxr-xr-x`, что позволяет другим пользователям ходить с правами экскурсантов по вашим каталогам. Но ничего в них не трогать и не записывать.

Но многие пользователи хотят держать других пользователей подальше от своих файлов. Установив права доступа файла, `-rw-------` вы никому не покажете этот файл и не дадите записать в него. Также хорошо закрывает от всех файлы защита соответствующего каталога `drwx------` .

3.9.4 Изменение прав доступа

Команда `chmod` используется для установки (изменения) прав доступа файла. Только владелец файла может менять права доступа к нему.

Синтаксис команды имеет вид:

```
chmod a,u,g,o+,-r,w,x filenames
```

Кратко, вы выбираете из **a**ll (все), **u**ser (пользователь), **g**roup (группа) или **o**ther (другие). Далее указываете, либо вы добавляете права (+), либо лишаете прав (-). И наконец, вы указываете один или несколько режимов: **r**ead, **w**rite или **e**xecute. Несколько примеров допустимых команд:

chmod a+r stuff

Дает всем пользователям право читать файл stuff.

chmod +r stuff

То же самое, что и ранее (a - по умолчанию).

chmod og-x stuff

Лишает права на выполнение всех, кроме владельца.

chmod u+rwx stuff

Разрешает владельцу все (read, write и execute).

chmod o-rwx stuff

Запрещает все (read, write и execute) пользователям категории другие (other).

3.10 Управление связями файлов

Связи позволяют давать одному физическому файлу много имен. Системой файлы распознаются по **индексам файлов**, которые являются уникальными идентификаторами в рамках системы.[22] На самом деле каталог - это перечень индексов файлов с соответствующими этим индексам номерами. Каждое имя файла в каталоге привязано к конкретному индексу.

3.10.1 Жесткие связи

Команда `ln` используется для создания множества связей для одного файла. Например, скажем, что у вас есть файл `foo`. Используя `ls -i` можно посмотреть индекс этого файла.

---

[22] Команда `ls -i` выдаст вам индексы файлов.

```
# ls -i foo
22192 foo
#
```

Здесь файл `foo` имеет в файловой системе индекс 22192. Мы можем создать новую связь для этого файла под именем bar:

```
# ln foo bar
```

С помощью `ls -i` можно убедиться, что оба файла имеют один и тот же индекс.

```
# ls -i foo bar
22192 bar   22192 foo
#
```

Теперь, обращаясь к `foo` или `bar` мы фактически обратимся к одном у тому же файлу. Поэтому, если мы меняем что-то в файле `foo`, эти же самые изменения произойдут в файле `bar`.

Эти связи известны, как *жесткие связи* (*hard links*), поскольку они реализуются прямой ссылкой на индекс файла. Обратите внимание, что в рамках одной файловой системы вы можете организовать только жесткие связи; символические связи (смотрите ниже) не имеют этого ограничения.

Когда вы удаляете файл командой `rm`, на самом деле вы удаляете только одну ссылку на файл. Если вы введете команду

```
# rm foo
```

Удалаяется только связь, имеющая имя `foo`; `bar` будет как и прежде существовать. Файл только тогда действительно удаляется, когда на него больше нет связей. Обычно файлы имеют только одну связь, так что команда rm действительно приведет к удалению файла. Однако, если файл имеет много ссылок, применение rm приведет только к удалению одной связи; для того, чтобы удалить файл, вы должны удалить все связи на этот файл.

Команда `ls -l` покажет число ссылок на файл (кроме прочей информации)

```
# ls -l foo bar
-rw-r--r--   2 root     root          12 Aug  5 16:51 bar
-rw-r--r--   2 root     root          12 Aug  5 16:50 foo
#
```

Вторая колонка с цифрой “2” показывает число связей файла.

Самом деле оказывается, что каталоги представляют из себя справочник типа ”имена-индексы”. Кроме прочего, каждый каталог имеет минимум две жесткие ссылки: “`.`” (ссылка, указывающая на самого себя) и “`..`” (ссылка, указывающая на родительский каталог). В корневом каталоге (`/`) ссылка “`..`” указывает на сам же каталог `/`.

3.10.2 Символические связи

Символические связи, это другой тип связей, отличающийся от жестких связей. Символические связи позволяют давать новые имена файлам, но при этом не ссылаются на индекс файла.

Команда `ln -s` создаст символическую ссылку на указанный файл. Например, если мы воспользуемся командой

```
# ln -s foo bar
```

мы создадим символьную ссылку `bar`, указывающую на файл `foo`. Если теперь используем команду `ls -i`, то увидим, что два файла имеют различные индексы.

```
# ls -i foo bar
22195 bar   22192 foo
#
```

Однако, используя `ls -l`, мы видим, что файл `bar` имеет символический указатель на `foo`.

```
# ls -l foo bar
lrwxrwxrwx   1 root     root          3 Aug  5 16:51  bar - foo
-rw-r--r--   1 root     root         12 Aug  5 16:50 foo
#
```

При символической ссылке не используются биты прав доступа (они всегда отображаются, как `rwxrwxrwx`). Вместо этого, права доступа к файлу, полученному символической ссылкой, определяются правами доступа к файлу, на который он ссылается (в нашем примере определяется правами файла `foo`).

Функционально, жесткие ссылки и символические ссылки похожи, но есть некоторые различия. Например, вы можете создать символическую ссылку на файл, который не существует; так нельзя сделать применительно к жесткой ссылке. Символические ссылки обрабатываются ядром иным образом, чем жесткие. Это скорее техническое отличие, но иногда важное. Символические ссылки полезны, поскольку они позволяют идентифицировать файл, на который они указывают; для жестких ссылок нет простого способа определить, какие файлы привязаны к одному и тому же индексу.

Ссылки используются во многих местах системы Linux. Символические ссылки особенно важны для образов разделяемых библиотек в `/lib`. Смотри дополнительную информацию в Разделе 4.7.2.

3.11 Управление работами

3.11.1 Работы и процессы

Управление работами (**job control**) это возможность, которую предоставляют многие оболочки, включая (Bash и Tcsh). Управление работами[23] позволяет управлять множеством команд или **работ** одновременно. Прежде, чем вы закопаетесь значительно глубже, следует поговорить о **процессах**.

Каждый раз, когда вы выполняете программу, вы начинаете то, что известно, как *процесс*. Процесс - это название для выполняемой программы. Команда `ps` выдает перечень имеющих место в данный момент процессов. Вот пример:

```
/home/larry# ps

  PID TT STAT  TIME COMMAND
   24  3 S     0:03 (bash)
  161  3 R     0:00 ps

/home/larry#
```

PID (**Process IDentificator**), перечисленные в первой колонке, это неповторяющиеся числа приписанные всем идущим процессам.

Последний столбец (COMMAND) дает имя выполняемой команды. Здесь мы видим только процессы, которые инициировал Larry. (В системе выполняется и много других процессов. Команда “`ps -aux`” может выдать перечень всех происходящих в данный момент процессов).

В выведенном перечне указаны `bash` (это оболочка, используемая Larry) и сама команда `ps`. Как вы видите, bash выполняется параллельно с командой `ps`. bash выполнит `ps`, когда Larry введет команду. После окончания `ps` (после того, как выдана таблица

---

[23] (прим. переводчика: job - работа в добрые старые времена страшноватых IBM/360 переводилось как ”задание”, но лучше это не тащить в сегодня)

процессов), управление возвращается к процессу `bash`, который выдает на экран подсказк готовности к приему новых команд.

Выполняемый процесс известен shell как *работа*. Термины *процесс* и *работа* взаимозаменяемы. Однако процесс обычно воспринимается, как "работа", когда речь идет об **управлении работами** (**job control**)- свойстве shell, позволяющем уделять внимание нескольким независимым работам.

В большинстве случаев пользователи выполняют в каждый момент времени одну работу, ту которая соответствует последней переданной shell команде. Однако, используя управление работами, вы можете одновременно выполнять несколько работ, по необходимости переключаясь с одной на другую. Какая от этого польза? Давайте предположим, что вы редактируете текстовый файл и неожиданно хотите прерваться и сделать что-то другое. С помощью управления работами вы можете отложить редактирование и, вернувшись к подсказке shell, начать какую-то другую работу. После этого вы можете вернуться к редактированию, именно к тому месту, где вы прервали редактирование. Это всего один пример. Управление работами очень полезно на практике.

3.11.2 Выполнение работ на переднем плане и в фоне

Работы могут выполняться как на **переднем плане**, так и в **фоне**. На переднем плане в каждый момент может быть только одна работа. Работа переднего плана, это работа, с которой вы взаимодействуете, она получает информацию с клавиатуры и посылает результаты на ваш экран. (Кроме, разумеется, случаев, когда вы сами перенаправляете вход или выход, как описывалось в Разделе 3.8). С другой стороны, фоновые работы не получают информации с терминала, в общем случае они тихо (в смысле - мирно) выполняются, не испытывая потребности в общении с пользователем.

Некоторые работы требуют очень большого времени для своего завершения и не свершают ничего внешне интересного в процессе этой работы. Компиляция программ - одна из таких работ, как и компрессия больших файлов. Нет вразумительных причин, почему вы должны при этом сидеть рядом и мучительно ждать, когда эти работы закончатся. Вы можете просто запустить их в фоне. Пока они там выполняются, вы можете заняться другими программами.

Работы могут быть также **отложены**. Отложенная работа - это работа, которая в данный момент не выполняется и временно остановлена. После того, как вы остановили работу, в дальнейшем вы можете ее продолжить как на переднем плане, так и в фоне. Возобновление приостановленной работы не изменит ее состояния - при возобновлении она начнется с того места, на котором была приостановлена.

Имейте в виду, что приостановка работы, это не *прерывание* работы. Когда вы прерываете идущий процесс (нажимая клавиши прерывания, обычно это $ctrl - C$), то убиваете процесс насовсем. (Клавиши прерывания можно переустанавливать командой stty. По умолчанию прерывание находится под $ctrl - C$, но мы не можем это гарантировать для всех систем). Если работа убита, то уж убита, и нет другого способа возобновить ее, как вновь запустить сначала, используя прежнюю команду. Заметим также, что некоторые программы могут перехватывать прерывания, тогда нажатие $ctrl - C$ не приведет к немедленному прекращению работы. Это позволит программе выполнить необходимые операции аккуратного завершения. Некоторые программы вообще не позволят вам их прервать.

3.11.3 Работа в фоне и ликвидация работ

Давайте начнем с простого примера. Команда `yes` - вроде бы бесполезная команда, посылающая бесконечный поток "y" на стандартный выход. (Но это очень полезно. Если вы направите через конвейер эти "y" на вход другой команды, которая требует ответов `yes` и "no" на вопросы, поток "y" даст подтверждение на все вопросы). Попробуйте.

```
/home/larry# yes
y
y
y
y
y
```

Это закончится в *бесконечности.* Вы можете убить процесс, нажав клавиши прерывания; обычно это $ctrl - C$. Чтобы нас больше не раздражал поток нескончаемых "y", перенаправим его в `/dev/null`. Как вы помните, `/dev/null` выступает в качестве "черной дыры" для данных. В ней исчезают бесследно любые данные.

```
/home/larry# yes  /dev/null
```

Ох, теперь намного лучше. Ничего не печатается, но и подсказка shell не появляется. Это потому, что программа продолжает работать, посылать "y" в `/dev/null`. Снова нажмите клавиши прерывания, чтобы прекратить это.

Давайте предположим, что мы хотим, чтобы команда `yes` продолжала работать, но также хотим получить обратно подсказку shell, чтобы выполнять другие работы. Мы можем перевести команду `yes` в фоновый режим, что позволит ей выполняться, но без выхода на взаимодействие с пользователем.

Чтобы переместить процесс в фоновый режим, необходимо после команды символ “&”.

```
/home/larry# yes  /dev/null &;
[1] 164
/home/larry#
```

Вы видите, что мы вновь получили подсказку. Но что значит “1 164”? И выполняется ли команда `yes` на самом деле?

“1” представляет **номер работы** для программы `yes`. Shell приписывает номер каждой выполняемой работе. Поскольку ”yes” - одна единственная работа, которая в данный момент выполняется, ей присвоен номер 1. “164” - идентификатор процесса (PID); это номер, присвоенный системой работе. Любой из этих номеров можно использовать при обращении к работе, как это будет показано в дальнейшем.

Теперь мы имеем выполняемый процесс `yes` в фоновом режиме, непрерывно посылающий поток ”y”-ков в `/dev/null`. Чтобы проверить состояние этого процесса, используйте внутреннюю команду shell - `jobs`.

```
/home/larry# jobs
[1]+  Running                 yes /dev/null  &
/home/larry#
```

Ясно, что она выполняется. Вы можете также воспользоваться командой `ps`, показанной ранее, для проверки статуса работ.

Для завершения работы используйте команду `kill`. Эта команда может брать в качестве аргумента как номер работы, так и идентификатор процесса. Это была работа номер 1, так что используя команду

```
/home/larry# kill %1
```

мы ликвидируем работу. При идентификации работы по номеру необходимо впереди ставить символ процента (“%”).

Теперь, после ликвидации, мы можем снова использовать `jobs` для проверки:

```
/home/larry# jobs

[1]+  Terminated              yes /dev/null

/home/larry#
```

Работа действительно мертва, и если мы снова воспользуемся командой `jobs`, ничего не будет выведено на экран.

Вы можете также убить работу, используя номер идентификатора процесса (PID), который выводится наряду с работой, когда вы начинаете работу (в фоновом режиме). В нашем пример PID равен 164, так что команда

```
/home/larry# kill 164
```

эквивалентна

```
/home/larry# kill %1
```

Вам не надо использовать "%", когда вы обращаетесь к работе по номеру идентификатора процесса.

3.11.4 Остановка и возобновление работы

Есть другой способ перевести работу в фоновый режим. Вы можете начать работу нормально (в режиме переднего плана), **остановить** работу и продолжить в фоновом режиме.

Сначала начнем работу "нормально":

```
/home/larry# yes  /dev/null
```

Поскольку опять работа выполняется на переднем плане, вы не получите обратно на экран подсказку shell.

Теперь, вместо того, чтобы прерывать работу с помощью *ctrl−C*, мы остановим работу. *Приостановка* работы не убивает ее. Чтобы осуществить приостановку работы, надо нажать соответствующие клавиши, обычно это *ctrl − Z*.

```
/home/larry# yes  /dev/null
[ctrl-Z]
[1]+  Stopped                yes /dev/null
/home/larry#
```

Пока работа остановлена, она просто не выполняется. На нее не тратится время процессора. Но вы всегда можете возобновить работу, и она продолжится как ни в чем не бывало.

Для возобновления работы в режиме переднего плана используйте команду `fg` ("foreground" - передний план).

```
/home/larry# fg
yes /dev/null
```

Shell снова выдаст на экран имя команды, чтобы вы могли проконтролировать, какую работу вы активизировали в режиме переднего плана. Вновь остановите работу с помощью $ctrl - Z$. В этот раз используйте команду `bg` (“background” - задний план, фоновый режим), чтобы перевести работу в фоновый режим. Эффект будет аналогичен тому, как если бы вы набрали после команды “`&`”.

```
/home/larry# bg
[1]+ yes /dev/null &
/home/larry#
```

И мы получили назад подсказку. Команда jobs сообщит, что команда yes действительно выполняется, и мы можем снова ее убить с помощью команды `kill`, как мы это уже делали.

Как теперь остановить работу? Использование $ctrl - Z$ не поможет, поскольку работа находится в фоновом режиме. Ответ - переместить работу на передний план, а затем остановить. Вы можете использовать `fg` как для остановленных работ, так и для работ, находящихся в фоне.

Существует большая разница между фоновой работой и остановленной. Остановленная работа не выполняется и не использует время процессора, да и никакой работы, честно говоря, в этот момент не делает (но занимает память, хотя по воле свопинга может оказаться на диске). Работа в фоновом режиме и выполняется, и занимает память. Она может даже выводить что-то на экран, хотя это может раздражать вас, когда вы работаете над чем-то другим. Например, если вы использовали команду:

```
/home/larry# yes &
```

без перенаправления stdout в `/dev/null`, поток ”y” будет выводиться на экран без возможности прервать это (вы не сможете использовать $ctrl - C$ для прерывания работ фонового режима). Чтобы остановить эту бесконечную выдачу, вам следует использовать команду fg для перевода работы в режим переднего плана, а затем использовать $ctrl - C$, чтобы ее убить.

Еще одно замечание. Команды `fg` и `bg` обычно переводят на передний план или в фоновый режим работы, которые были остановлены последними (что определяется символом “`+`” после номера работы, это когда вы используете команду jobs). Если вы выполняете много работ одновременно, вы можете перевести на передний план или, наоборот, в фоновый режим конкретную работу заданием идентификатора работы в качестве аргумента команд fg или bg, как в

```
/home/larry# fg %2
```

(перевод на передний план работы номер 2) или

```
/home/larry# bg %3
```

(перевод в фон работы номер 3).

Для этих команд нельзя использовать идентификаторы процессов. Кроме того, использование только номеров работ, как в

```
/home/larry# %2

 эквивалентно

/home/larry# fg %2
```

Помните, что управление работами, это свойство shell. Команды `fg`, `bg` и `jobs` - внутренние команды shell. Если по какой-то причине вы используете shell, который не поддерживает управление работами, там вы не найдете этих команд.

В дополнение к этому, есть некоторые аспекты управления работами, которые различаются в Bash и Tcsh. Некоторые оболочки не имеют управления работами, хотя большинство оболочек Linux имеют такую возможность.

3.12 Использование редактора vi

Текстовый редактор, это программа, используемая для редактирования файлов, которые содержат текст, например письма, С-программы или системные конфигурационные файлы. Хотя в Linux много всяких разных редакторов, единственный редактор, который вы с гарантией найдете в любом UNIX - это `vi` (“visual editor”). `vi` - это не самый простой в использовании редактор. Но поскольку он так распространен в мире UNIX и в любой момент может вам потребоваться, он заслуживает хоть какого-то описания здесь.

Выбор редактора, это дело персонального вкуса и стиля. Многие пользователи предпочитают витиеватый и мощный *Emacs* - редактор с самым большим набором возможностей, по сравнению со всеми другими редакторами в мире UNIX. Например, Emacs имеет свой собственный встроенный диалект языка программирования LISP и множество расширений (одно из которых “Eliza”- в некотором роде программа искусственного интеллекта). Однако, поскольку Emacs со всеми поддерживающими его файлами сравнительно велик, его нет на многих системах. `vi`, наоборот, маленький и удаленький, но, увы, более сложный в использовании. Но когда вы с ним освоитесь, вы поймете, что он очень простой. Правда осваивать его сложно.

Этот раздел - вразумительное введение в `vi`. Мы не будем обсуждать все его свойства, а только те, которые вы должны знать, чтобы начать работать. Если вы пожелаете знать больше деталей, обратитесь к страницам Руководства.

3.12.1 Концепции

При использовании `vi` в любое время вы можете находиться в одном из трех режимов работы. Эти режимы известны как командный режим, режим вставки и режим последней строки.

Когда вы начинаете работать с `vi` - вы в командном режиме. Этот режим позволяет использовать определенные команды для редактирования файлов или перехода в другие режимы. Например, напечатав “`x`” при нахождении в командном режиме, удаляете символ, находящийся перед курсором. Стрелки передвигают курсор по редактируемому файлу. Большинство команд, используемых в командном режиме, состоит из одного или двух символов.

Вставку или редактирование текста вы осуществляете в режиме вставки. При использовании `vi` вы, возможно, большую часть времени находитесь именно в этом режиме. Вы переходите в режим вставки с помощью команды “`i`” (“insert” - вставка) из командного режима. В режиме вставки вы вставляете текст в документ на место, указываемое курсором. Для завершения режима вставки и возврата в командный режим следует нажать *esc*.

Режим последней строки - это специальный режим, используемый для расширения возможностей командного режима. При вводе таких команд они появляются в последней строке экрана. Например, если вы напечатаете “:” в командном режиме, вы перейдете в режим последней строки и сможете использовать такие команды, как “`wq`” (записать (write) файл и выйти (quit) из vi), или “`q!`” (выйти из vi без сохранения изменений). Режим последней строки в общем случае используется для команд `vi`, которые длиннее одного символа. В режиме последней строки вы вводите однострочные команды и нажимаете *enter* для их выполнения.

3.12.2 Начала vi

Лучший способ освоить эту концепцию, это вызвать `vi` и отредактировать файл. В примере “`screens`”, приводимом ниже, мы собираемся только показать несколько строк текста, будто бы экран состоит всего из шести строк (вместо двадцати четырех).

```
          Вызов vi
```

```
vi filename
```

где <`filename`> - имя редактируемого файла.

Ну так вызовите `vi`, напечатав

```
/home/larry# vi test
```

для редактирования файла `test`. Вы увидите нечто вроде

```
|~                                                                  |
|~                                                                   |
|~                                                                   |
|~                                                                   |
|~                                                                   |
|”test”[Newfile]|
```

Столбец символов “ ” говорит о том, что вы стоите на конце файла.

3.12.3 Вставка текста

Вы находитесь в командном режиме; для того, чтобы вставлять текст в файл, нажмите *i* (что переведет вас в режим вставки) и начинайте печатать.

```
|Now is the time for all good men to come to the aid of the party. |
|~                                                                   |
|~                                                                   |
|~                                                                   |
|~                                                                   |
|~|
```

При вставке текста вы можете напечатать столько строк, сколько пожелаете (нажимая *return* после каждой строки), и можете корректировать ошибки используя клавишу возврата (backspace).

Для завершения режима вставки и возврата в командный режим нажмите *esc*.

В командном режиме вы можете использовать клавиши со стрелками для перемещения по файлу. Здесь, поскольку мы имеем только одну строку текста, попытки использовать стрелки ”вверх” и ”вниз” приведут лишь к тому, что vi на вас загудит.

Есть несколько способов вставки текста, отличных от использования команды `i`. Например, команда `a` вставляет в текст, начиная *после* текущего положения курсора, вместо текущей позиции курсора. Используйте левую стрелку для перемещения курсора между словами “good” и “men”.

```
|Now is the time for all goodmen to come to the aid of the party.  |
|~                                                                  |
|~                                                                  |
|~                                                                  |
|~                                                                  |
|~|
```

Нажмите *a*, для начала режима вставки, напечатайте “`wo`”, а затем нажмите *esc* для возврата в командный режим.

```
|Now is the time for all good women to come to the aid of the party.|
|~                                                                  |
|~                                                                  |
|~                                                                  |
|~                                                                  |
|~|
```

Для того, чтобы начать вставку текста в строку ниже текущей, используйте команду “`o`”. Например, нажмите *o* и напечатайте строчку или две

```
|Now is the time for all good women to come to the aid of the party.|
|Afterwards, we´ll go out for pizza and beer.                      |
|~                                                                  |
|~                                                                  |
|~                                                                  |
|~|
```

Но помните, что в любое время вы находитесь либо в командном режиме (где команды, такие как `i`, `a` или `o` могут применяться) или в режиме вставки (где вы вставляете текст, а затем с помощью *esc* возвращаетесь в командный режим) или в режим последней строки (в котором вы расширяете расширяемые команды, как это обсуждается ниже).

3.12.4 Удаление текста

В командном режиме команда ”`x`” удаляет символ перед курсором. Если вы нажмете *x* пять раз, вы закончите в ситуации:

```
|Now is the time for all good women to come to the aid of the party.|
|Afterwards, we'll go out for pizza and                             |
|~                                                                  |
|~                                                                  |
|~                                                                  |
|~|
```

Теперь нажмите *a*, вставьте некоторый текст, а затем нажмите *esc*:

```
|Now is the time for all good women to come to the aid of the party.|
|Afterwards, we'll go out for pizza and Diet Coke.                  |
|~                                                                  |
|~                                                                  |
|~                                                                  |
|~|
```

Вы можете удалять целые строки, набирая команду `dd` (т.е. нажимая *d* дважды). Если ваш курсор на второй строке, и вы напечатали `dd`,

```
|Now is the time for all good women to come to the aid of the party.|
|~                                                                  |
|~                                                                  |
|~                                                                  |
|~                                                                  |
|~|
```

Чтобы удалить слово, на котором находится курсор, используйте команду `dw`. Поместите курсор на слово “good” и напечатайте `dw`.

```
|Now is the time for all women to come to the aid of the party.     |
|~                                                                  |
|~                                                                  |
|~                                                                  |
|~                                                                  |
|~|
```

3.12.5 Изменение текста

Вы можете заменить фрагменты текста, используя команду `R`. Поместите курсор на первую букву слова “party”, нажмите *R* и напечатайте слово “hungry”.

```
|Now is the time for all women to come to the aid of the hungry.   |
|~                                                                  |
|~                                                                  |
|~                                                                  |
|~                                                                  |
|~|
```

Использование `R` для редактирования текста очень походит на на команды `i` и `a`, но `R` заменяет прежний текст вместо вставки в него. Команда `r` заменяет один символ, отмеченный курсором. Например, переместите курсор на начало слова “Now” и напечатайте `r`, а следом `C`, то вы получите:

```
|Cow is the time for all women to come to the aid of the hungry.   |
|~                                                                  |
|~                                                                  |
|~                                                                  |
|~                                                                  |
|~|
```

Команда “~” изменяет размер буквы, отмеченной курсором: большую делает маленькой и наоборот. Например, если вы поместите курсор на “o” в “Cow” и затем последовательно будете нажимать , вы в конечном итоге получите:

```
|COW IS THE TIME FOR ALL WOMEN TO COME TO THE AID OF THE HUNGRY.    |
|~                                                                  |
|~                                                                  |
|~                                                                  |
|~                                                                  |
|~|
```

3.12.6 Команды перемещения

Вы уже знаете, как использовать стрелки для перемещений по документу. Вы также можете использовать команды `h`, `j`, `k`, и `l` для перемещения курсора влево, вниз, вверх и вправо соответственно. Это удобно, если (по каким-то причинам) ваши клавиши со стрелками не работают как надо.

Команда `w` перемещает курсор на начало следующего слова; `b` - перемещает на начало предыдущего слова.

Команда `0` (это ноль) передвигает курсор на начало текущей строки, а команда `$` перемещает на конец строки.

При редактировании больших файлов вы хотите перемещаться вперед и назад сразу на размер экрана. Нажатием $ctrl - F$ курсор перемещается на экран вперед, с помощью $ctrl - B$ - на экран назад.

Для того, чтобы переместить курсор в конец файла, напечатайте `G`. Можно переместиться также на любую строку, напечатав команду `10G` вы переместите курсор на десятую строку файла. Для того, чтобы встать на начало (на первую строку), используйте `1G`.

Вы можете сочетать команды перемещения с другими командами, такими как удаление. Например, команда `d$` удалить от местоположения курсора до конца строки; `dG` удалит все от курсора до конца файла и т.д.

3.12.7 Сохранение файлов и выход из vi

Для выхода из `vi` без внесения изменений в ранее существовавший файл используйте команду `:q!`.

Когда вы напечатаете “`:`”, курсор переместится на последнюю строку экрана, поскольку вы перейдете в режим последней строки.

```
|COW IS THE TIME FOR ALL WOMEN TO COME TO THE AID OF THE HUNGRY.   |
|~                                                                 |
|~                                                                 |
|~                                                                 |
|~                                                                 |
|~                                                                 |
|:|
```

В режиме последней строки могут выполняться некоторые расширенные команды. Одна из них - `q!`, которая позволяет выйти из `vi` без записи. Команда `:wq` сохраняет (записывает) файл, а затем выходит из `vi`. Команда `ZZ` (в режиме команд, без ‘`:`”) эквивалентна `:wq`. Помните, что вы должны нажать *enter* после набора команды в режиме последней строки. Если хотите записать файл без выхода из `/vi`, используйте просто `:w`.

3.12.8 Редактирование еще одного файла

Для того, чтобы отредактировать другой файл, используйте команду `:e`. Например, чтобы прекратить редактирование файла `test` и перейти к редактированию файла `foo`, используйте команду

```
|COW IS THE TIME FOR ALL WOMEN TO COME TO THE AID OF THE HUNGRY.   |
|~                                                                 |
|~                                                                 |
|~                                                                 |
|~                                                                 |
|~                                                                 |
|:e foo|
```

Если вы используете :e без предварительного сохранения файла, то сначала вы получите сообщение об ошибке.

```
|Nowritesincelastchange(”:edit!”overrides)|
```

которое просто означает, что `vi` не желает редактировать другой файл, пока не будет сохранен первый. В этот момент вы можете использовать `:w`, чтобы сохранить исходный файл, а затем использовать `:e` или использовать команду

```
|COW IS THE TIME FOR ALL WOMEN TO COME TO THE AID OF THE HUNGRY.   |
|~                                                                 |
|~                                                                 |
|~                                                                 |
|~                                                                 |
|~                                                                 |
|:e! foo|
```

“`!`” говорит `vi`, что вы на самом деле имеете в виду - редактировать новый файл без сохранения изменений, которые делались в первом.

3.12.9 Включение других файлов

Если вы используете команду `:r`, вы можете включить содержимое другого файла в текущий файл. Например, команда

```
:r foo.txt
```

вставит содержимое файла `foo.txt` в данное место текста.

3.12.10 Выполнение команд Shell

Вы можете также выполнять команды прямо из `vi`. Команда `:r!` работает как `:r`, но вместо чтения файла она вставляет выход данной команды в буфер, в место, где находится курсор. Например, если вы используете команду

```
:r! ls -F
```

вы получите в результате

```
|COW IS THE TIME FOR ALL WOMEN TO COME TO THE AID OF THE HUNGRY.   |
|letters/                                                          |
|misc/                                                             |
|papers/                                                          |
|~                                                                 |
|~|
```

Вы можете выполнить команду `a`, находясь в редакторе `vi` и вернуться в редактор после ее завершения Напрмиер, если вы используете команду

```
:! ls -F
```

будет выполнена команда `ls -F`, а результат выдан на экран, а не вставлен в редактируемый файл. Если вы используете команду

```
:shell
```

`vi` запустит shell, который позволит временно ”отложить” vi и выполнить команды. После выхода из shell (используя команду `exit`) вы вернетесь в `vi`.

3.12.11 Получение помощи

`vi` не слишком силен в интерактивной помощи (да и большинство UNIX-ов также), но вы всегда можете посмотреть страницы Руководства для `vi`. `vi` - это ”визуальная составляющая” редактора `ex`; это `ex` делает многое для поддержания режима последней строки и командного режима в `vi`. Так что в дополнение к чтению Руководства по `vi` посмотрите также Руководство по `ex`.

3.13 Установка среды

Shell обеспечивает различные механизмы настройки вашей рабочей среды. Мы уже упоминали ранее, что shell больше, чем команда интерпретации - это также мощный язык программирования. Но обсуждение программирования на shell увело бы нас далеко в сторону, а мы бы хотели познакомить вас с некоторыми способами упрощения вашей работы в UNIX за счет использования некоторых дополнительных полезных свойств shell.

Как мы упоминали ранее, различные оболочки используют различный синтаксис для написания своих программ. Например, Tcsh использует синтаксис, похожий на язык Си, в то время как shell Баурна имеет другой синтаксис. В этом разделе мы не будем заниматься их различиями, а рассмотрим примеры, используя синтаксис shell Баурна[24]

3.13.1 Сценарии shell

прим.переводчика :
применительнокпрограммамэтоготипаванглоязычнойлитературепоследнеевремяпреимущественно
"сценарий", хотято, чтоподэтимимеетсяввидувомногихкнигахнарусскомназывается"традиционно"

Предположим, что вы часто используете серию команд и хотели бы сократить об'ем постоянной печати за счет группировки их в одну команду. Например, команды

```
/home/larry# cat chapter1 chapter2 chapter3  book
/home/larry# wc -l book
/home/larry# lp book
```

об'единяют файлы, содержащие главы книги: `chapter1`, `chapter2`, `chapter3` и помещают результат в файл `book`. Затем подсчитывается число строк в книге (в файле `book`) и отображается на дисплее и, наконец, печатается командой `lp`.

Вместо введения каждый раз этих команд, вы можете собрать их в один сценарий (командный файл). Сценарии shell мы кратко опишем в Разделе 3.13.1. А сценарий, который выполнит вышеприведенные команды, будет выглядеть следующим образом

```
#!/bin/sh
# A shell script to create and print the book

cat chapter1 chapter2 chapter3  book
wc -l book
lp book
```

Если этот сценарий будет помещен в файл `makebook`, то вы можете просто использовать далее команду

```
/home/larry# makebook
```

[24] (прим. переводчикf: как все обычно и делают).

которая выполнит все команды сценария. Сценарии shell - это обычные текстовые файлы, которые вы можете создавать с помощью редактора вроде `emacs` или `vi`. (`vi` обсуждался в Разделе 3.12)

Давайте посмотрим на этот сценарий. Первая строка "`#!`bin/sh/" говорит о том, что этот файл есть сценарий и сообщает shell, как выполнить сценарий. В данном случае необходимо передать сценарий для выполнения команде bin/sh/, где bin/sh/ - сама программа shell. Почему это важно? В большинстве систем UNIX bin/sh/ - это shell Баурновского типа, например `bash`. Инициируя работу сценария shell выполняется, используя bin/sh/, при этом мы гарантируем, что сценарий будет выполняться именно под shell Баурновского типа (а не, скажем, под C shell). Этот сценарий будет выполняться под shell Баурна, даже если вы используете Tcsh (или какой-то другой C shell) как свою рабочую оболочку.

Вторая строка представляет из себя комментарий. Комментарии начинаются символом "`#`" и могут продолжаться до конца строки - они игнорируются shell и могут использоваться программистом для пояснений.

Остальные строки сценария - обычные команды в том виде, в каком бы вы их вводили прямо на выполнение. Shell читает каждую строку сценария и выполняет эту строку, как будто вы ввели эту строку в ответ на подсказку shell.

Права доступа важны для сценариев. Если вы создали сценарий, вы должны убедиться, что вы имеете права на его выполнение.[25] Можно использовать команду

```
/home/larry# chmod u+x makebook
```

чтобы дать самому себе разрешение на выполнение shell-сценария `makebook`.

3.13.2 Перемещение shell и среда

Shell позволяет определять **переменные**, как и большинство языков программирования. Переменная - это порция данных, которой дано имя.[26]

ВНИМАНИЕ! Имейте в виду, что Tcsh, также, как и C shell, используют различные механизмы определения переменных, отличающиеся от используемых здесь. Здесь обсуждается shell Баурна. Когда вы присвоите значение переменной (используя

---

[25] Если вы создавали сценарий в редакторе, то он (обычно) не получает автоматически прав на выполнение.

[26] (прим. переводчика: В языке shell переменные не определяются (в традиционном смысле), так как все они одного типа - "строкового", речь может идти только об их инициировании: присваивании начальных значений).

оператор ``=``), вы сможете получить это значение, добавив перед именем переменной символ ``$``, как это показано ниже

```
/home/larry# foo=``hello there´´
```

Переменной foo присвоено значение ``hello there``. Теперь вы можете обратиться к этой переменной, добавив перед именем символ ``$``. Команда

```
/home/larry# echo $foo
hello there
/home/larry#
```

дает тот же самый результат, что и

```
/home/larry# echo ``hello there´´
hello there
/home/larry#
```

Эти переменные являются внутренними для shell. Это означает, что только shell имеет доступ к этим переменным. Это может быть полезно для сценариев; если вам надо сохранить информацию о имени файла, вы, например, можете поместить его в переменную. Команда set может показать вам перечень всех определенных переменных shell.

Shell позволяет **экспортировать** переменные в среду. **Среда** - это множество переменных, к которым могут иметь доступ все выполняемые команды. Определив однажды переменную внутри shell[27] командой `export` вы можете передать ее среде.

ВНИМАНИЕ! Здесь вновь есть отличие между Bash и Tcsh. При использовании Tcsh используется другой сиснтаксис для помещения переменных в среду (используется команда `setenv`). Дополнительную информацию можно найти в Руководстве по Tcsh.

Среда очень важна в системах UNIX. Она позволяет конфигурировать некоторые команды за счет установки переменных, о которых знают команды.

Вот небольшой пример. Переменная среды `PAGER` используется командой man. Она указывает команду, которая используется в свою очередь командой man для просмотра Руководства на экране. Если вы установите в качестве значения `PAGER` имя другой команды, то эта команда вместо будет обеспечивать просмотр вместо more (которая применялась по умолчанию).

Присвойте `PAGER` значение ``cat``. Выдача на экран руководства будет вся разом, а не поэкранно, как это делала команда more.

---

[27] (прим. переводчика: определить - здесь означает "присвоить значение"),

```
/home/larry# PAGER=cat
```

Теперь экспортируйте `PAGER` в среду.

```
/home/larry# export PAGER
```

Попробуйте команду `man ls`. Руководство промелькнет по вашему экрану без (желательных) задержек.

Теперь, если присвоить `PAGER` значение “`more`”, то для выдачи на экран будет использоваться команда more.

```
/home/larry# PAGER=more
```

Обратим внимание на то, что нам не надо заново использовать команду `export` после изменения значения `PAGER`. Необходимо только раз экспортировать переменную; любые изменения, которые будут происходить после этого, будут отражаться в среде.

Страницы Руководства для конкретных команд содержат информацию о том, использует ли команда какие-то переменные среды. Например, Руководство по команде man говорит о том, что для определения режима выдачи страницы руководства на экран используется переменная `PAGER`. Некоторые команды совместно используют переменные среды, например, многие команды используют переменную среды `EDITOR` для указания используемого редактора.

Переменные среды используются также для сохранения важной информации о процедуре входа. Например переменная `HOME` содержит имя вашего домашнего каталога.

```
/home/larry/papers# echo $HOME
/home/larry
```

Другая интересная переменная среды - `PS1`, которая определяет главную подсказкн shell. Например,

```
/home/larry# PS1=``Your command, please:  ´´
Your command, please:
```

Для переустановки подсказки обратно в нормальное состояние (когда она показывает текущий рабочий каталог, после которого следует значек “`#`”), выполните следующее:

```
Your command, please:  PS1=``w# ´´
/home/larry#
```

В Руководстве `bash` есть подробное описание синтаксиса, используемого при установке подсказки.

Переменная среды PATH Когда вы используете команду `ls`, как shell находит соответствующий выполняемый файл (программу) для `ls`? На самом деле в большинстве систем `ls` находится в `/bin/ls`. shell использует переменную среды `PATH` ("ТРОПА") для указания возможного местоположения выполняемых файлов соответствующих команд.

Например, ваша переменная`PATH` может иметь значение

```
/bin:/usr/bin:/usr/local/bin:.
```

Это список каталогов (в которых shell будет искать команду), отделяемых друг от друга двоеточием "`:`". Когда вы используете команду `ls`, `shell` прежде всего проверяет наличие `/bin/ls`, затем `/usr/bin/ls` и т.д.

Обратите внимание на то, что переменная `PATH` не помогает находить обычные файлы. Например, если вы используете команду

```
/home/larry# cp foo bar
```

shell не использует `PATH` для нахождения местопребывания файлов `foo` и `bar` - предполагается, что эти имена однозначно определяют место. shell использует `PATH` только для нахождения команды cp.

Это экономит вам массу времени; это означает, что вы не обязаны помнить, где находятся выполняемые файлы команд. Во многих системах выполняемые файлы разбросаны во многих местах, таких как `/usr/bin`, `/bin` или `/usr/local/bin`. Вместо того, чтобы писать полное имя команды (вроде `/usr/bin/cp`), вы просто указываете в `PATH` перечень каталогов, которые бы вы хотели, чтобы shell автоматически просматривал.

Обратите внимание, что `PATH` содержит "`.`", что означает "текущий рабочий каталог". Это позволяет вам создавать shell-сценарии или программы и выполнять их как команды из текущего каталога, без необходимости указывать это прямо (как в случае `./makebook`). Если каталог не указан в `PATH`, то shell не будет его просматривать в поиске команд, это касается и текущего каталога.

3.13.3 Shell-Сценарии инициализации

В дополнение к shell-сценариям, которые создаете вы, существует множество сценариев, которые использует сам shell для своих целей. Наиболее важными среди них являются **сценарии инициализации**, которые автоматически выполняются shell при вашем входе в систему.

Сценарии инициализации сами по себе - это обычные сценарии, как это описывалось выше. Но они очень полезны при установке вышей среды путем автоматического выполнения набора команд при вашем входе в систему. Например, если вы всегда используете команду `mail` для проверки своей почты в момент входа в систему, вы можете поместить эту команду в свой сценарий инициализации и она будет выполнена автоматически.

Как Bash, так и Tcsh делают различие между начальным shell (вызываемым при входе в систему) и прочими вызовами shell. Начальный shell вызывается в момент входа пользователя в систему; часто это единственный экземпляр shell, который вы используете. Но если вы вызываете shell из другой программы, такой как`vi`, вы тем самым запускаете новый (экземпляр) shell. Кроме того, когда вы запускаете на выполнение shell-сценарии, вы автоматически инициируете новый экземпляр shell.

Файлы инициализации, используемые в Bash: `/etc/profile` (устанавливается системным администратором, выполняется всеми экземплярами начальных пользовательских `bash`, вызванными при входе пользователей в систему), `$HOME/.bash_profile` (выполняется при входе пользователя) и `$HOME/.bashrc` (выполняемый всеми прочими неначальными экземплярами `bash`). Если `.bash_profile` отсутствует, вместо него используется `.profile`.

Tcsh использует следующие сценарии инициализции: `/etc/csh.login` (выполняется всеми пользовательскими `tcsh` в момент входа в систему), `$HOME/.tcshrc` (выполняется во время входа в систему и всеми новыми экземплярами `tcsh`) и `$HOME/.login` (выполняется во время входа после `.tcshrc`). Если `.tcshrc` отсутствует, вместо него используется `.cshrc`.

Для того, чтобы лучше понять функции этих файлов, вам следует больше узнать о shell. Программирование на shell сложный вопрос, далеко выходящий за рамки этой книги. Дополнительную информацию можно получить из Руководства на `bash` и `tcsh`.

3.14 Не хотите ли отправиться в самостоятельное плавание

Надеемся, что мы дали достаточно информации относительно того, как использовать систему. Имейте в виду, что большая часть интересных и важных аспектов Linux здесь не обсуждалась - здесь рассматривались только самые базовые. Но этот фундамент поможет вам быстро освоить и использовать более сложные приложения. Если все рассмотренное здесь вам не показалось волнующе интересным, не унывайте - в Linux есть еще много того, с чем следовало бы познакомиться. Один из незаменимых способов изучения системы - это чтение Руководства. Хотя многие страницы Руководства могут выглядеть и достаточно сложными, если вы будете достаточно глубоко копать, вы откопаете несметные россыпи полезной информации.

Мы также советуем прочитать какую-то полную книгу по системе UNIX. В UNIX значительно больше разнообразных возможностей, чем можно увидеть с певого взгляда - к сожалению, они выходят за рамки этой книги. Некоторые хорошие книги по UNIX указаны в Приложении А.

4 Администрирование

Эта глава представляет обзор функций администрирования системы Linux, включая ряд особых функций, предназначенных исключительно для администратора системы.

Как в каждой бочке меда присутствует ложка дегтя, так и каждая система имеет своего администратора. А администрирование системы - это очень важная и иногда пожирающая уйму времени работа, даже если вы единственный пользователь системы.

Мы постараемся обсудить здесь наиболее важные вещи, связанные с администрированием, о котором вы должны знать при использовании Linux, чтобы не испытывали неудобств при работе с ОС. Чтобы быть не слишком болтливыми и приятными собеседниками, мы и раньше рассматривали только основные черты, пропуская многие важные детали. Вам следует прочитать книгу " Linux System Administrator's Guide", если у вас относительно Linux серьезные намерения. Это поможет вам лучше понять как там все происходит, и как там все взаимодействует. В крайнем случае, стоит все это просмотреть, чтобы знать что в книге содержится и какой помощи вам следует от нее ожидать.

4.1 О корнях власти, волшебной шапке и приятных ощущениях.

Как вы знаете, UNIX различает различных пользователей, так что то, что они могут сделать друг другу и системе, регулируется (например, не хочется, чтобы кто-то читал чужие любовные письма). Каждый пользователь получает **account** (регистрируется в системе), что включает имя пользователя, домашний каталог и т.д. В дополнение к регистрации реальных людей, регистрируются (для них также открывается счет :-) несколько специальных пользователей, имеющих привилегии. Наиболее "важный" даже среди них пользователь - `root` (корень).

4.1.1 Регистрация root

Обычные пользователи в общем случае ограничены так, что они не могут причинить вред кому-либо другому в системе (включая саму систему), кроме самих себя. Права доступа к файлам в системе организованы таким образом, что простой пользователь не может удалить или изменить файл, файл в каталогах, которые пользователи используют совместно (такие как `/bin` и `/usr/bin`). Большинство пользователей также защищают свои собственные файлы так, что не могут их изменить, а иногда и вообще добраться до них.

Но все эти ограничения не распространяются на пользователя root. Пользователь root может читать, модифицировать или удалять любой файл системы, изменять его права доступа или менять его владельца. Он (`root`) может также выполнять специальные (привилегированные) программы, такие как разбитие диска на разделы или создание файловой системы. Основная идея состоит в том, что некто (их может быть несколько), кто выполняет регистрацию пользователей (и носит имя `root`), должен, когда это необходимо, иметь возможность выполнять работы, которые не могут быть выполнены обычным рядовым пользователем. Поскольку root может делать все, что угодно, ему легко совершить какую-то ошибку, приводящую к катастрофическим последствиям.

Например, если вы как обычный пользователь случайно попытаетесь удалить файл в `/etc`, система не разрешит вам это сделать. Но, если вы вошли как `root`, система даже не пикнет, выполняя все, что прикажете. Легко уничтожить систему, пребывая в системе в качестве root. Лучший способ избежать неприятностей, это:

- Посидеть на собственных ладошках, прежде чем нажать *return* для выполнения команды, которая может быть причиной катастрофы. Например, если вы собираетесь очистить каталог, перед нажатием *return* перечитайте всю команду и убедитесь, что она написана правильно.

- Не привыкайте использовать `root`. Чем более комфортно вам будет в роли root, тем больше вы будете путать ваши привилегии с привилегиями нормального пользователя. Например, вы можете подумать, что вы сейчас находитесь в системе как `larry`, хотя на самом деле будете неудержимым root.

- Используйте отличающуюся подсказку для `root`. Для этого следует внести изменения в root-овский `.bashrc` или `.login` файл для того, чтобы сделать подсказку для root отличной от других. Например, многие используют символ ‘‘`$`’’ в подсказках обычных пользователей и оставляют символ ‘‘`#`’’ для подсказки root.

- Входите под именем `root` только тогда, когда это абсолютно необходимо. И, как только вы закончите работу root-а, выйдите (выведите root-а из системы). Чем меньше используете root, тем меньше навредите системе.

Разумеется, есть племя хакеров, которые используют `root` практически всегда и везде. Но каждый из них когда-то по глупости уничтожил хотя бы (в лучшем случае) одну систему. Есть общее правило: пока вы не познакомились с неограниченными возможностями `root`, и не привыкли к отсутствию ограничений, входите под root в крайнем случае.

Разумеется, все совершают ошибки. Однажды сам Linus Torvalds (создатель linux) случайно удалил все поддерево каталогов, содержавшее программы ядра. Многие часы работы пропали (бы) в один миг навсегда. К счастью, однако, благодаря своему знанию кодов файловой системы, он смог перезагрузить систему и реконструировать дерево каталогов вручную.

Давайте по-другому, если вы представите использование `root` как ношение специальной волшебной шапки, которая дает вам могущество, так что вы можете мановением руки разрушить целые города, то уместная мысль, что надо очень следить за своими руками. А поскольку такая мощь опасна (да и рукам неудобно), лучше без большой нужды не надевать волшебную шапку, даже если в шапке у вас повышается самоуважение.

4.1.2 Злоупотребление системой

С приходом ощущения власти приходит желание вредить. Это темная сторона администрирования в UNIX, но всякий через это когда-то должен пройти. Большинство пользователей UNIX никогда не получат возможность испытать это на университетских и производственных системах UNIX. Только высокооплачиваемые и высокообразованные системные администраторы могут входит в систему под именем `root`. Действительно, во многих таких заведениях пароль root - это строго охраняемый секрет. Это священная корова фирмы. Много делается попыток пролезть под именем root в систему; она представляется мудрой и устрашающей силой, покоряющейся только тем, кто знает заклинания.

Такая позиция по отношению к `root` очень легко приводит к опасностям и соблазнам. Поскольку `root` столь одурманивающая штука, то когда пользователь дорывается до возможности войти под root, прослеживается начало использования свалившихся привилегий в плане вредительства. Я знавал таких "системных администраторов", которые читали без разрешения почту других пользователей и вообще вели себя как дети, которым дали столь мощную клевую "игрушку".

Поскольку `root` имеет в системе такие привилегии, требуется определенный уровень зрелости и самоконтроля, чтобы использовать этот account (этот привилегированный "счет"), как это было задумано - для эксплуатации системы. Существует негласный закон чести в отношениях администратора с пользователями. Как вы будете себя чувствовать, если системный администратор читает ваши письма и просматривает ваши файлы? До сих пор нет достаточно серьезной юридической основы для неприкосновенности личной информации в многопользовательских компьютерных системах. В системах семейства UNIX пользователь root имеет возможность преодолевать все штатные механизмы защиты системы. Важно, чтобы у администратора были доверительные отношения с пользователями системы. Невозможно переоценить важность этого.

4.1.3 Взаимодействие с пользователями

Безопасность UNIX довольно рыхлая от рождения. Вопросы безопасности были додуманы "в догонку" - исходно система создавалась в неформальной атмосфере, когда все вмешивались в работу друг друга. Благодаря этому, даже несмотря на меры безопасности, у нормального пользователя существуют возможности причинить системе вред.

Системный администратор может выбрать две тактики взаимодействия с злоупотребляющими[28] пользователями. Это может быть параноидная тактика и тактика доверия. Системный администратор с паранойей обычно своими действиями наносит больше вреда, чем предотвращает. Одна из моих любимых присказок: "Никогда не списывай на зловредность то, что можно списать на тупость". Взгляните с другой стороны, большинство пользователей не имеют возможностей и знаний, чтобы причинить реальный вред системе. 90% процентов того,

---

[28] (прим. переводчика: в исходном, а не в узко русском смысле этого слова)

что делает пользователь, причиняя вред системе (например, забивая пользовательский раздел огромными файлами или выполняя сразу несколько экземпляров громадной программы), он делает просто не подозревая, что он кому-то создает проблемы. Мне приходилось сталкиваться с пользователями, которые были источниками огромных неприятностей, но они они действовали по простоте душевной, а не со зла.

Когда вы имеете дело с пользователями, которые опасны потенциально, не накидывайтесь на них с обвинениями. Старое правило "презумпции невиновности" все еще не отменили. Лучше всего поговорить с пользователем, поспрашивать о его проблемах, вместо того, чтобы идти на конфронтацию. Самое плохое, это пытаться отвечать ему "встречными" неприятностями. Это создаст вокруг вас - системного администратора - много подозрений, поставит под сомнение вашу способность корректно сопровождать систему. Если пользователь решит, что вы не верите ему или даже не любите, он может обвинить вас в том, что вы удаляете его файлы и вообще подсматриваете. Вряд ли вы хотите оказаться в такой ситуации.

Если вы убедились, что пользователь действительно пытается ‘‘взломать’’ систему или умышленно ей вредит, старайтесь не отвечать угрозами на угрозы. Вместо этого просто предупредите его, но сохраняйте гибкость. Во многих случаях вы можете"схватить его за руку" в процессе свершения вредительства - вот тут и предупредите. Скажите, чтобы он так больше не делал. Но если вы снова его поймаете на вредительстве, то убедитесь, что это действительно намеренно. Я просто не смогу перечислить все случаи, когда оказывалось, что неприятность была либо случайной, либо я сам был виноват.

4.1.4 Установление правил

Лучший способ управления системой - это управление без применения железного кулака. Может так вы хорошо управляли в армии, но это не для UNIX. Имеет смысл сделать простой и гибкий свод руководств для пользователей, но чем меньше у вас будет правил, тем меньше шансов их нарушить. Даже если ваши правила использования системы очень ясны и разумны, пользователи все равно время от времени будут их без злого умысла нарушать. Это, в особенности, относится к новичкам в UNIX, которые еще только изучают основы системы. Да и вы сами можете время от времени рассылать гигабайтные файлы всем пользователям системы... Пользователям надо помочь понять правила и об’яснить, зачем они нужны.

Если вы создали руководство для пользователей системы, убедитесь, что причины введения тех или иных правил им понятны. Если вы этого не сделаете, пользователи творчески подойдут к тому, как обходить эти правила. может быть и не сознавая, что они их действительно обходят.

4.1.5 Что все это значит

Мы не можем до последней детали расписать вам, как эксплуатировать систему. Большая часть философии зависит от того, как вы используете систему. Если у вас много пользователей, то это сильно отличается от того, когда их мало, или вообще вы один. Но при любом раскладе очень полезно задуматься, что в данной конкретной системе дествительно означают слова "системный администратор" (или "администратор системы").

Должность администратора системы не делает вас крутым юниксистом. На свете много системных администраторов, которые мало что знают о UNIX. Похоже, что существует много "нормальных" пользователей, которые, знают о UNIX больше любого системного администратора. Пребывание в должности администратора не дает вам права использовать угрозы в адрес пользователей. Именно потому, что система дает вам привилегию устроить из файлов пользователя все, что угодно, вы не имеете никакого права это делать.

Наконец, быть системным администратором, это невесть что. При этом не имеет значения, опекаете вы маленький 386-ой или суперкомпьютер Cray. Знание заветного пароля root не принесет вам денег и славы; оно поможет сопровождать систему и поддерживать ее работоспособность. Вот так.

4.2 Загрузка системы

Существует несколько способов загрузки системы: либо с дискеты, либо с жесткого диска.

4.2.1 Использование загрузочной дискеты

Многие загружают Linux используя ‘‘загрузочную дискету’’, которая содержит копию ядра Linux. В ядре есть информация о корневом разделе Linux, так что ядро знает, где искать на жестком диске корневую файловую систему. (Команда rdev может использоваться для установки корневого раздела в образе ядра; см. ниже). Это тип дискеты, созданной, например, Slackware в процессе инсталляции.

Для создания своей собственной загрузочной дискеты, сначала разместите образ ядра на своем жестком диске. Оно должно быть в файле `/Image` или `/etc/Image`. Некоторые инсталляции используют для формирования ядра файл `/vmlinux`.

Вместо этого у вас может быть скомпрессированное ядро. Скомпрессированное ядро само раскомпрессируется при загрузке в память и занимает значительно меньше места на диске. Если у вас есть скомпрессированное ядро, оно находится в файле `/zImage` или `/etc/zImage`.

Зная, где у вас находится ядро, установите корневое устройство в образе ядра на имя вашего корневого раздела командой `rdev`. Формат команды:

```
rdev kernel-name root-device
```

где <`kernel-name`>; это имя образа ядра, а <`root-device`> - имя корневого раздела Linux. Например, для установки корневого устройства в ядре `/etc/Image` на `/dev/hda2` используется команда

```
# rdev /etc/Image /dev/hda2
```

rdev может устанавливать другие опции в ядре, такие как взятый по умолчанию режим SVGA, для использования во время загрузки. Используйте “`rdev -h`” для получения помощи.

После установки корневого устройства вы можете просто скопировать образ ядра на дискету. При копировании данных на дискету, хорошо бы сначала отформатировать дискету в MS-DOS. При форматировании выдается информация о секторах и треках дискеты, так что можно определить какую плотность записи имеет эта дискета.

Например, для копирование файла ядра `/etc/Image` на дискету в `/etc/fd0` используйте команду

```
# cp /etc/Image /dev/fd0
```

Теперь эта дискета должна загружать Linux.

4.2.2 Использование LILO

Другой метод загрузки - это использование LILO, программы, которая располагается в загрузочном секторе вашего жесткого диска. Эта программа выполняется, когда система загружается с жесткого диска и может автоматически загрузить Linux из образа ядра, хранящегося на жестком диске.

LILO может быть также использована, как начальный загрузчик для нескольких операционных систем, позволяя вам выбирать во время загрузки, какую операционную систему (например, Linux или MS-DOS) загружать. Когда вы загружаетесь с использованием LILO, то загружается операционная система, установленная по умолчанию, если вы не нажмете *ctrl*, *alt* или *shift* во время выполнения загрузочной последовательности. Если вы нажмете любой из этих ключей, то вам будет выдана подсказка загрузчика, в ответ на которую вы напечатаете имя операционной системы, которую надо загрузить (например, ``linux'' или ``msdos''). Если вы нажмете *tab* в ответ на подсказку загрузчика, вам будет выдан перечень доступных операционных систем.

Простой способ инсталлировать LILO - отредактировать файл конфигурации `/etc/lilo.conf` и выполнить команду

```
# /sbin/lilo
```

Файл конфигурации LILO содержит ``stanza''("стансы" - не пугайтесь, это действительно про поэзию). для каждой операционной системы, которую вы желаете загрузить. Лучший способ продемонстрировать это на примере конфигурационного файла LILO config. Нижеприведенные установки для системы, которая имеет корневой раздел Linux на `/dev/hda1` и раздел MS-DOS на `/dev/hda2`.

```
# Tell LILO to modify the boot record on /dev/hda (the first
# non-SCSI hard drive). If you boot from a drive other than /dev/hda,
# change the following line.
boot = /dev/hda

#  Name  of  the  boot  loader.  No  reason  to  modify  this
#  un less you're doing some serious hacking on LILO.
install = /boot/boot.b

# Have LILO perform some optimization.
compact

# Stanza for Linux root partition on /dev/hda1.
image = /etc/Image   # Location of kernel
   label = linux     # Name of OS (for the LILO boot menu)
   root = /dev/hda1  # Location of root partition
   vga   =   ask     # Tell kernel to ask for SVGA modes at boot time

# Stanza for MSDOS partition on /dev/hda2.
other = /dev/hda2    # Location of partition
   table = /dev/hda  # Location of partition table for /dev/hda2
   label = msdos     # Name of OS (for boot menu)
```

Стансы первой операционной системы в файле `config` - это та ОС, которую LILO загружает по умолчанию. Вы можете выбрать другую ОС во время загрузки в ответ на подсказку LILO, как это уже обсуждалось ранее.

Помните, что каждый раз, когда вы изменяете образ ядра на диске, вы должны заново выполнить `/sbin/lilo`, чтобы изменения отразились в загрузочном секторе вашего диска.

Имейте также в виду, что если вы используете здесь строку ``root ='', нет смысла использовать rdev для установки корневого раздела в образе ядра. LILO установит ее во время загрузки.

Linux FAQ (смотри Приложение A) дает дополнительную информацию по тому, как использовать LILO при загрузке Boot Manager OS/2.[29]

4.3 Выключение системы

Выключение Linux - это немножко акробатика. Не забывайте, что никогда нельзя просто выключить питание или нажать кнопку "reset" во время работы системы. Ядро отслеживает диск при вводе- выводе с помощью буферов. Если вы перезагружаете систему, не дав шанса ядру переписать буфера на диск, вы можете попортить файловые системы.

Необходимы и другие меры предосторожности при выключении. Всем процессам посылается сигнал, который позволяет им красиво умереть (записав, что надо и закрыв все файлы и т.д.). Файловые системы для безопасности размонтируются. Если вы желаете, система может также предупредить пользователей, что предстоит выключение, чтобы дать им шанс тоже (красиво) выйти из системы.

Простейший способ выключения, это использование команды `shutdown`. Формат команды

```
shutdown time warning-message
```

<`time`> - время выключения системы (в формате hh:mm:ss - чч:мм:сс) и <`warning-message`> - сообщение, выдаваемое на терминалы всех пользователей перед выключением. Вы можете просто указать время (<`time`>) как ``now'', что приведет к безотлагательному выключению. Опция `-r` приведет к перезагрузке после выключения.

Например, выключить систему в 8:00 вечера можно командой

```
# shutdown -r 20:00
```

Команда `halt` может инициировать немедленное выключение без посылки предупреждающих сообщений или предоставления паузы перед выключением. `halt`

---

[29] (прим. переводчика: *The Linux FAQ* - Часто Задаваемые Вопросы по Linux.

полезна, если вы единственный пользователь системы и хотите выключить систему и вырубить питание.

ВНИМАНИЕ! На выключайте электропитание и не перезагружайте ее, пока не увидите на консоли сообщение:

```
The system is halted
```

Важно сделать выключение "чисто", используя команды `shutdown` или `halt`. В некоторых системах нажатие *ctrl − alt − del* будет перехвачено системой и приведет к ее выключению, но в других системах использование "затычки для вулкана" приведет к немедленной перезагрузке системы и может быть причиной неприятностей.

4.4 Работа с пользователями

Вне зависимости от того, много у вас пользователей или нет, важно понять проблему работы с пользователем Linux. Даже если вы единственный пользователь вы должны иметь различные account для `root` и для себя. Каждый человек, использующий систему, должен иметь свой собственный account (быть индивидуально зарегистрированным в системе). Редко может быть целесообразно, чтобы несколько человек входили в систему под одним именем. Здесь дело не только в безопасности, но account используется для идентификации пользователя в системе. Необходимо иметь возможность проследить, кто что делает.

4.4.1 Концепция работы с пользователями

Система сохраняет различную информацию о каждом пользователе К такого рода информации относится перечисленная ниже.

username

уникальный идентификатор, присваиваемый каждому пользователю в системе. Примеры имен пользователей `larry`, `karl` и `mdw`. Могут использоваться буквы и цифры, а также нижнее подчеркивание и точка. Обычно имена пользователей ограничиваются восемью символами.

user

ID (или UID) - идентификатор пользователя - уникальный номер, присваиваемый каждому пользователю системы. Система обычно отслеживает инентификаторы пользователей, а не имена.

group

ID (или GID) - идентификатор группы это идентификатор группы пользователя. В Разделе 3.9 мы обсуждали права группы; каждый пользователь принадлежит к одной или более группам, определенных системным администратором. Подробнее об этом ниже.

password

Система также хранит в зашифрованном виде пароль пользователя. Команда passwd используется для установки и изменения пароля.

full name

"**Полное имя**" или "**действительное имя**" хранится вместе с именем пользователя. Например, пользователь schmoj может иметь действительное имя "Joe Schmo"[30]

home directory

Домашний каталог - это каталог, в который пользователь начально попадает при входе в систему. Каждый пользователь должен иметь свой собственный домашний каталог, обычно ниже `/home`.

login shell

Исходный shell - это shell, который запускается для пользователя при его входе в систему. Это, например, может быть `/bin/bash` и `/bin/tcsh`.

Файл `/etc/passwd` содержит эту информацию про пользователей. Каждая строка этого файла содержит информацию об одном пользователе; формат строки имеет вид:

```
username:encryptedpassword:UID:GID:fullname:homedirectory:loginshell
```

Например, это может выглядеть так:

```
kiwi:Xv8Q981g71oKK:102:100:Laura Poole:/home/kiwi:/bin/bash
```

Как видно, первое поле "`kiwi`" - имя пользователя. Следующее поле "`Xv8Q981g71oKK`" - зашифрованный пароль. Пароли в читаемом виде в системе не хранятся. Сами пароли шифруются как секретные ключи. Другими словами, вы должны знать пароль, чтобы его расшифровать. Эта форма шифрации достаточно надежна.

Некоторые системы Linux используют "теневой пароль", в котором информация о паролях хранится в файле `/etc/shadow`. Поскольку `/etc/passwd` всем доступен,

---

[30] (прим. переводчика: неужели для английского уха оно звучит также красиво, как для русского).

`/etc/shadow` обеспечивает дополнительный уровень секретности своей недоступностью. Теневой пароль обеспечивает и некоторые другие свойства, вроде прекращения действия пароля и т.д.; мы здесь не будем вдаваться в эти тонкости.

Третье поле, “`102`”, - идентификатор пользователя (UID). Оно должно быть уникальным для каждого пользователя. Четвертое поле, “`100`”, идентификатор группы (GID). Этот пользователь принадлежит к группе номер 100. Информация по группе хранится в файле `/etc/group`. Смотрите дополнительную информацию в Разделе 4.4.5.

Пятое поле - полное имя пользователя , “Laura Poole”. Последние два поля - домашний каталог пользователя (`/home/kiwi`) и исходный shell (`/bin/bash`) соответственно. Домашний каталог пользователя не обязательно должен иметь имя, совпадающее с именем пользователя (username). Однако это помогает в идентификации.

4.4.2 Добавление пользователей

При добавлении пользователя следует совершить несколько шагов. Первое, пользователь должен быть занесен в файл паролей `/etc/passwd` под уникальным именем и идентификатором. Должны быть описаны идентификатор группы (GID), полное имя и другая информация.

Должен быть создан домашний каталог пользователя и установлены необходимые права доступа. Домашний каталог должен быть снабжен необходимыми файлами инициализации shell. Должны быть выполнены и другие работы по конфигурации (например, создан spool для входной почты).

Хотя очень несложно добавлять пользователей вручную (я так делаю), когда вы сопровождаете систему со многими пользователями, легко что-то упустить. Самый простой способ регистрации пользователей - это использование диалоговой программы, которая задаст вам все необходимые вопросы и автоматически скорректирует все необходимые системные файлы. Эта программа называется `useradd` или `adduser`, в зависимости от вашего дистрибутива. Страницы руководства для этих программ должны быть достаточно понятными.

4.4.3 Удаление пользователей

Аналогично, удаление пользователей может быть выполнено с помощью команд `userdel` или `deluser` в зависимости от конкретного дистрибутива.

Если вы пожелаете временно ”отключть” пользователя от системы, (без удаления его account), вы можете просто приписать звездочку (“`*`”) в поле пароля в файле пароля

`/etc/passwd`. Например, изменить у `kiwi` в файле `/etc/passwd`

```
kiwi:*Xv8Q981g71oKK:102:100:Laura Poole:/home/kiwi:/bin/bash
```

это закроет для `kiwi` вход в систему.

4.4.4 Занесение атрибутов пользователя

После создания пользователя вам может потребоваться изменить его атрибуты, такие как домашний каталог и пароль. Простейший способ - это прямо изменить значения в `/etc/passwd`. Чтобы установить пароль пользователя используйте команду passwd.

Например,

```
# passwd larry
```

изменить пароль larry.

Только `root` может изменять пароли других пользователей. Пользователи также могут изменять пароли с помощью команды passwd, но только свои собственные.

В некоторых системах имеются команды `chfn` и `chsh`, позволяющие пользователю самому устанавливать свое полное имя и исходные атрибуты shell. Если нет, то пользователи должны просить системного администратора изменить эти атрибуты для них.

4.4.5 Группы

Как мы говорили, каждый пользователь принадлежит к одной или более группам. Единственное значение группы замыкается на права доступа к файлу, как вы помните из Раздела 3.9. Каждый файл имеет "групповое владение" (“group ownership”), то есть хранит права доступа, которые определяют, как члены группы могут обращаться с файлом.

Существует несколько групп, определяемых системой, вроде `bin`, `mail` и `sys`. Пользователи не могут принадлежать к какой-либо из этих групп. Эти группы используются для системных файлов. А пользователи, наоборот, принадлежат к специальной группе, например users. Если вам хочется больше проблем, вы можете поддерживать несколько групп пользователей, например `student`, `staff` и `faculty`.

Файл `/etc/group` содержит информацию о группах. Формат каждой строки следующий

```
group name:password:GID:other members
```

Примерами групп могут быть:

```
root:*:0:
users:*:100:mdw,larry
guest:*:200:
other:*:250:kiwi
```

Первая группа, `root`, специальная системная группа зарезервированная для root. Следующая группа, `users`, для обычных пользователей. Она имеет идентификатор группы (GID) 100. К этой группе имеют доступ пользователи `mdw` и `larry`. Помните, что в `/etc/passwd` каждый пользователь получил (выдаваемый по умолчанию) GID. Но пользователь может принадлежать более, чем к одной группе, путем добавления имен пользователей в другие группы (в строки файла `/etc/group`). Команда `groups` перечисляет, в какие группы вы входите.

Третья группа, guest, для гостей, а other для "других" пользователей. Пользователь `kiwi` имеет доступ и в эту группу.

Как вы можете видеть, поле "пароль" в `/etc/group` редко используется. Иногда оно используется для установления пароля на доступ к группе. Это редко бывает нужно. Для защиты привилегированных групп от обычных пользователей (с помощью команды `newgroup`) установите в поле пароля звездочку ("*").

Команды `addgroup` или `groupadd` могут быть использованы для добавления групп в вашу систему. Обычно легче просто самому добавить запись в `/etc/group`, поскольку не требуется других настроек при добавлении группы. Для удаления группы просто удалите соответствующую запись в `/etc/group`.

4.5 Архивация и компрессирование файлов

Прежде, чем мы сможем говорить о сохранении (резервировании) программ, мы должны представить инструменты, используемые для архивации файлов и программ в системах UNIX.

4.5.1 Использование tar

Команда `tar` наиболее часто используется для архивации файлов. Формат команды `tar`

```
tar options file1 ... fileN
```

где <`options`> есть список команд и опций для `tar`, а <`file1`> ... <`fileN`> есть список файлов добавляемых в архив или извлекаемых из него.

Например, команда

```
# tar cvf backup.tar /etc
```

упакует все файлы, содержащиеся в `/etc`, в архив `tar` под именем `backup.tar`. Первый аргумент команды `tar` -``` ``cvf' ```, это (внутренняя) ”команда” `tar`. “`c`” указывает `tar` создать новый архивный файл. Опция “`v`” заставляет tar выводить имя каждого архивируемого файла. Опция “`f`” говорит, что следующий аргумент - `backup.tar` - имя созданного архивного файла. Остальные аргументы команды `tar` - имя файла и имя добавляемого в архив каталога.

Команда

```
# tar xvf backup.tar
```

извлечет архивный файл в текущий каталог. Это может быть иногда и небезопасным занятием, когда извлеченные из архива файлы затирают существовавшие файлы.

Поэтому перед извлечением архивированных файлов важно знать, где файлы следует распаковать. Например, вы заархивировали следующие файлы: `/etc/hosts`, `/etc/group` и `/etc/passwd`. Если вы используете команду

```
# tar cvf backup.tar /etc/hosts /etc/group /etc/passwd
```

в начало имени каждого файла добавится каталог с именем `/etc`. Чтобы извлечь файлы в нужное место, вам потребуется использовать следующие команды:

```
# cd /
# tar xvf backup.tar
```

поскольку файлы извлечены с сохраненной в архиве тропой

Если вы заархивировали файлы командой

```
# cd /etc
# tar cvf hosts group passwd
```

имя каталога не сохраняется в архивном файле. Поэтому вы должны выполнить “`cd /etc`” перед извлечением файлов. Вы обратили внимание: то, как вы создали архивный файл сильно влияет на то, в каком месте его следует извлекать. Команда

```
# tar tvf backup.tar
```

может быть использована для просмотра ”индекса” архивного файла перед его распаковкой. Таким образом вы можете посмотреть, к какому каталогу относятся архивированные файлы и сможете извлечь файлы из архива в нужном месте.

4.5.2 gzip и compress

В отличие от архивирующих программ для MS-DOS, `tar` не компрессирует автоматически файлы в процессе архивирования. Поэтому, если вы архивируете два одно-мегабайтных файла, результирующий архивный файл будет размером два мегабайта. Команда `gzip` может использоваться для компрессирования файла (компрессируемый файл не обязан быть `tar`-файлом)

Команда

```
# gzip -9 backup.tar
```

скомпрессирует `backup.tar` и оставит вас наедине с `backup.tar.gz`, скомпрессированной версией файла. Опция `-9` говорит команде `gzip`, что следует использовать максимальную возможную компрессию.

Команда `gunzip` может быть использована для раскомпрессирования "зазипованного" файла. С аналогичным эффектом вы можете использовать "`gzip -d`".

`gzip` - сравнительно новый инструмент в кругах, приближенных к UNIX. Долгие годы вместо этого использовалась команда `compress`. Однако, по нескольким причинам (тут и патентные дрязги относительно алгоритма, и то, что `gzip` значительно эффективнее) `compress` оказался не у дел.

Обработанные командой `compress` файлы заканчивались расширением `.Z`. Например, `backup.tar.Z` - это компрессированная версия файла `backup.tar`, а `backup.tar.gz` - зазипованная версия.[31]

Команда `uncompress` используется для развертывания файла, который был обработан командой `compress`. Но команда `gunzip` тоже знает, как обращаться с такими файлами.

4.5.3 Можно вместе

Чтобы заархивировать и скомпрессировать группу файлов, вы можете использовать команды:

```
# tar cvf backup.tar /etc
# gzip -9 backup.tar
```

Результат будет `backup.tar.gz`. Для распаковки этого файла используйте обратную последовательность команд:

---

[31] (Чтобы еще надежнее запутать дело, для обозначения зазипованных файлов некоторое время использовалось расширение `.z` (маленькая "z"). В настоящее время (прим. переводчика: это для современников автора) официальное расширение - .gz.

```
# gunzip backup.tar.gz
# tar xvf backup.tar
```

Разумеется, всегда следует убедиться перед распаковкой файла, что вы в нужном каталоге.

Вы можете опереться на некоторую сообразительность UNIX, позволяющего сделать это одной командой[32]

```
# tar cvf - /etc | gzip -9c  backup.tar.gz
```

Здесь мы посылаем `tar`-файл, сформированный из `/etc`, в файл “`-`”, который представляет стандартный выход. Результат по конвейеру поступает на вход команды `gzip`, которая компрессирует этот файл и результат сохраняет в `backup.tar.gz`. Опция `-c` команды `gzip` говорит, что выход команды `gzip` посылает результат на стандартный выход, который перенаправляется на `backup.tar.gz`.

Единственная составная команда, используемая для распаковки этого архива, будет:

```
# gunzip -c backup.tar.gz | tar xvf -
```

Опять, команда `gunzip` раскомпрессирует содержимое файла `backup.tar.gz` и посылает результирующий файл на стандартный выход. Он по конвейеру передается команде `tar`, которая читает файл “`-`”, что в данном случае олицетворяет стандартный выход.

К счастью, команда `tar` также содержит опцию `z`, автоматически компрессируя-раскомпрессируя файлы, используя алгоритм компрессии `gzip`.

Например, команда (прим. переводчика: одна)

```
# tar cvfz backup.tar.gz /etc
```

эквивалентна

```
# tar cvf backup.tar /etc
# gzip backup.tar
```

Как и команда

```
# tar xvfz backup.tar.Z
```

может быть использована вместо

---

[32] (прим. переводчика: Верно, но сказать - одной ”командной строкой” - было бы честнее, да и проще описывать работу этой конструкции).

```
# uncompress backup.tar.Z
# tar xvf backup.tar
```

За дополнительной информацией обратитесь к Руководству по `tar` и `gzip`.

4.6 Использование дискет и осуществление резервирования

Дискеты часто используются как средство резервирования. Если у вас нет ленты (стриммера), можно использовать дискеты (хотя они медленнее и, в некотором смысле, менее надежны).

Вы можете использовать дискеты также для хранения отдельных файловых систем - в этом случае вы должны **монтировать** (**mount**) дискету для получеия доступа к ее данным.

4.5.4 Использование дискет для резервирования

Простейший способ резервирования на дискетах, это использование команды `tar`.

```
# tar cvfzM /dev/fd0 /
```

сделает полную копию вашей системы с использованием дисковода `/dev/fd0`. Опция “`M`” позволяет копировать на несколько дискет (multivolume backup); то есть, когда одна дискета заполнится, tar запросит следующую. Команда

```
# tar xvfzM /dev/fd0
```

может быть использована для полного восстановления. Этот метод может быть также использован для лент (`/dev/rmt0`).

Существует
несколько других программ для осуществления многотомного резервирования. Вам могут пригодиться программы ”`backflops`”, которые можно взять на `tsx-11.mit.edu`.

Создание полной копии системы может быть весьма время-ресурсо емким. Большинство системных администраторов использует ”инкрементальную” стратегию резервирования. Каждый месяц производится полное копирование, а каждую неделю - только тех файлов, которые были модифицированы в эту неделю. В этом случае, если вы грохнете свою систему в середине месяца, вы можете просто восстановить состояние на начало месяца, а затем восстановить понедельные изменения.

Команда `find` может быть полезна в выискивании файлов, которые изменились после какой-то даты. Несколько сценариев (командных файлов на shell) для инкрементального резервирования можно найти на `sunsite.unc.edu`.

4.5.5 Использование дискет в качестве файловых систем

Вы можете создать файловую систему на дискете точно также, как в разделе жесткого диска. Например,

```
# mke2fs /dev/fd0 1440
```

создает файловую систему на дискете на /dev/fd0. Размер файловой системы должен соответствовать размеру дискеты. Дискеты high-density 3.5" - размером в 1.44 Мбайт или 1440 блоков. Дискеты high-density 5.25" - размером в 1200 блоков.

Для того, чтобы иметь доступ к дискете, вы должны примонтировать содержащуюся на ней файловую систему. Команда

```
# mount -t ext2 /dev/fd0 /mnt
```

примонтирует дискету, находящуюся на `/dev/fd0` к каталогу `/mnt`. Теперь все файлы, находящиеся на дискете, будут находиться в каталоге `/mnt` вашего жесткого диска.[33] "`-t ext2`"указывает тип файловой системы (ext2fs). Если вы создали другой тип файловой системы на дискете, вам потребуется описать его тип команде `mount`.

"Точка монтирования" (каталог, к которому вы примонтируете файловую систему) должен существовать, когда вы применяете команду `mount`. Если он не существует, создайте его с помощью команды `mkdir` - и все проблемы.

Дополнительную информацию по файловым системам, монтированию и точкам монтирования смотрите в Разделе 4.8.

Важное замечание! Ввод/вывод на дискету буферизируется точно также, как и для жесткого диска. Когда вы меняете (достаете) дискету, вы не должны видеть горящую лампочку дисковода (пока ядро работает с буферами ввода/вывода). Важно, чтобы вы не извлекали дискету из дисковода до ее размонтирования, которое можно выполнить командой

```
# umount /dev/fd0
```

Нельзя просто взять и вытащить дискету, как в MS-DOS. При замене дискет сначала размонтируйте одну, а затем примонтируйте вторую.

---

[33] (прим. переводчика: в /mnt непосредственно будет находиться вершина дерева файловой системы дискеты, все остальное опосредовано - ниже по дереву).

4.6 Модернизация и инсталляция программ

Другая обязанность системного администратора - модернизация и инсталляция новых программ.

Сообщество приверженцев Linux очень динамично. Новые версии ядра появляются каждые несколько недель, да и другие программы изменяются не менее часто. Поэтому новые пользователи Linux часто чувствуют необходимость в постоянной модернизации (upgrade) своей системы, чтобы поспевать за изменениями, идущими лихой поступью. Это необходимо и это и потеря времени: отслеживать все изменения в мире Linux. Просто у вас может абсолютно все время уходить на модернизацию системы и лишь оставшееся - на собственно использование системы.

Ну, так когда желаете занятся модернизацией? Некоторые нутром чувствуют, что заниматься модернизацией пристало тогда, когда появилась новая версия дистрибутива, например, когда появляется новая версия Slackware. Многие пользователи Linux каждый раз при этом полностью переинсталлируют свою систему. Это тоже потеря времени. Обычно изменения от версии к версии Slackware незначительные. Бессмысленно переписывать и переинсталлировать 30 дисков, когда только 10% программ были действительно модифицированы.

Лучший вариант модернизации системы - это ручная работа: модернизируйте только те программные пакеты, про которые вы точно знаете, что их стоит менять. Это многих пугает: они хотят знать, что менять, и как, и что они теряют, если не модернизируют. Залог успеха в Linux - это преодолеть боязнь принципа "сделай сам", одного из фундаментальных принципов Linux.

Действительно, благостное состояние пользователя работающей и хорошо настроенной системы враз меняется при переинсталляции, поскольку, без сомнения, приводит и к перенастройке всего и вся, к тому, что опять все не работает, как это было при первой инсталляции системы. Так что определенные сеансы самопсихотерапии необходимы, чтобы иметь деловой настрой. Все, что требуется - это немножко "ноу-хау" по модернизации системы.

Вы обнаружите, что когда вы модернизируете одну компоненту вашей системы, другие вещи не должны ломаться. Например, большая часть моей системы оставлена со времен древней 0.96 MCC Interim installation. Тем не менее, я использую новейшую версию ядра и библиотек без проблем. Большей частью бессмысленно заниматься модернизациями, чтобы "не отстать от моды". Суета все это. Это вам не MS-DOS или Microsoft Windows. У нас нет серьезных причин обязательно работать на новейшей во все времена версии системы. Если вы осознаете, что вам действительно нужны некоторые вещи из новой версии - тогда модифицируйте на здоровье. А если нет, то лучше не надо. Другими словами модернизируйте только то, что надо, и только тогда, когда надо. Не модернизируйте во имя модернизации.

Наиболее важная часть вашей системы, как возможный об'ект модернизации, это ядро, библиотеки и компилятор gcc. Это три ключевые части вашей системы, и в некоторых случаях они бывают взаимозависимыми. Большая часть остального хозяйства вашей системы и без периодических модернизаций сойдет.

4.6.1 Модернизация ядра

Модернизация ядра - это просто надо взять исходные тексты и самому их откомпилировать. Вы должны компилировать ядро сами, поскольку вам решать, какие свойства включать и не включать, как и убедиться, что ядро будет оптимизировано применительно к вашей машине. Процесс вполне безболезненный.[34]

Исходные тексты ядра можно раздобыть на любом Linux-овском FTP-сервере (список смотрите в Разделе C). На `sunsite.unc.edu`, например, исходники ядро находятся в pub/Linux/kernel/. Версии ядра нумеруются с использованием номера версии ядра (kernel) и уровня исправления (patchlevel). Например, kernel version 0.99 patchlevel 11 обычно записывается как `0.99.pl11`, или еще проще `0.99.11`.

Исходники ядра распространяются в виде зазипованных tar-файлов. (Часто patch-файлы ("заплаты") выпускаются для текущей версии ядра, которая позволяет модернизировать исходники вашего действующего ядра на основе последнего уровня исправлений, используя программу patch. В большинстве случаев, между тем, обычно проще инсталлировать целиком новую версию ядра. Например, файл, содержащий исходники ядра 0.99.pl11 - `linux-0.99.11.tar.gz`.[35]

Распакуйте этот файл из каталога `/usr/src`; он создаст каталог `/usr/src/linux`, который содержит исходники ядра. Вам следует удалить или переименовать существующий `/usr/src/linux` перед распаковкой новой версии.

Когда исходники распакованы, вам необходимо убедиться, что две символические связи в `/usr/include` корректны. Для создания этих связей используйте команды

---

[34] (прим. переводчика: Настоящие парашютисты сами укладывают свой парашют).

[35] (прим. переводчика: На момент перевода книги существуют следующие серии ядер: 0.1-0.99, 1.0.1-1.0.9, 1.1.1-1.1.95, 1.2.1-1.2.13, 1.3.1-1.3.83. Ядра, принадлежащие серии с четной 2-й цифрой (1.0, 1.2), являются стабильными (то есть не включают никаких экспериментальных кодов). Ядра серии (1.1, 1.3) включают экспериментальные коды (такие как Mobile IP, IP-masquarading и т. п. в серии 1.3). Переводчик, исходя из декларированного автором принципа "от добра добра не ищут", работает в версии 1.2.13 и с нетерпением ждет появления версии 1.4.1 или 2.0.)

```
# ln -sf /usr/src/linux/include/linux /usr/include/linux
# ln -sf /usr/src/linux/include/asm /usr/include/asm
```

Если вы однажды создали эти связи, нет причины создавать их

снова, когда вы инсталлируете следующую версию ядра. (Смотрите Раздел 3.10 по поводу символических связей).

Обратите внимание, что для компиляции ядра у вас в системе должны быть инсталлированы компиляторы `gcc` и `g++` C и C++. Если вы нуждаетесь в более свежих версиях этих компиляторов, смотрите ниже Раздел 4.7.3.

Для компиляции ядра прежде всего выполните `cd` в `/usr/src/linux`. Выполните команду `make config`. Эта команда запросит у вас несколько настроечных опций, таких как "Какой тип файловой системы вы желаете включить в новое ядро". Затем, отредактируйте `/usr/src/linux/Makefile`. Убедитесь, что определение для `ROOT_DEV` корректно - оно определяет устройство, используемое в качестве корневой файловой системы во время загрузки. Обычное определение имеет вид:

```
ROOTDEV = CURRENT
```

Менять это нет смысла, кроме случая, когда вы меняете устройство для вашей корневой файловой системы.

Затем выполните команду `make dep` для отслеживания всех взаимосвязей исходных текстов Это очень выхный этап.

И наконец, вы готовы компилировать ядро. Команда `make Image` скомпилирует ядро и оставит образ нового ядра в файле `/usr/src/linux/Image`.[36] А команда `make zImage` скомпилирует скомпрессировнный образ ядра, который раскомпрессирует сам себя во время загрузки, а так занимает на диске меньше места.

После компиляции ядра вы должны либо скопировать его на загрузочную дискету (командой вроде "`cp Image /dev/fd0`"), либо инсталлировать его, используя LILO для загрузки с вашего жесткого диска. Дополнительную информацию можно найти в Разделе 4.2.2 .

4.6.2 Модернизация библиотек

Как говорилось ранее, большинство программ системы компилировалось для использования разделяемых библиотек содержащих общие подпрограммы, которыми

---

[36] (прим. переводчика: Начиная с ядер серии 1.2 собранное ядро записывается в каталог `/usr/src/linux/arch/i386/boot/Image` (если конечно вы собирали ядро для процессоров серии intel, а не alpha, mips или sparc.)

пользуются различные прикладные программы.

Если вы увидите сообщение

```
Incompatible library version
(Несовместимая версия библиотеки)
```

При попытке выполнить программу, вам необходимо модернизировать версию ваших библиотек, которые использует программа. Библиотеки совместимы в обратном направлении, то есть программа, откомпилированная для использования с более ранней версией библиотек, должна работать с новой версией библиотек. А обратное не справедливо.

Самая последняя версия библиотек может быть найдена FTP-серверах Linux. На `sunsite.unc.edu` они расположены в `/pub/Linux/GCC`. Файлы "версии" (“release”) должны описывать, какие файлы вам необходимо скачать, и как их инсталлировать. Кратко, вы должны иметь файлы `image-version.tar.gz` и `inc-version.tar.gz`, где версия указывает версию инсталлируемых библиотек, например 4.4.1.[37] Это зазипованные tar-файлы. Файлы образов содержат образы инсталлируемых библиотек в `/lib` and `/usr/lib`. Файл `inc` содержит include-файлы для инсталляции в `/usr/include`.

Файл `release-`*version*`.tar.gz` об'ясняет инсталляционную процедуру в деталях (конкретные инстукции для конкретных версий отличаются). В общем случае вы должны инсталлировать библиотечные `.a` и `.sa` файлы в `/usr/lib`. Эти библиотеки используются на этапе компиляции.

Дополнительно,
разделяемая библиотека образов файлов `libc.so.`*version* инсталлируется в `/lib`. Это разделяемые библиотеки образов загружаются во время выполнения использующими их программами. Каждая библиотека имеет символическую связь, использующую старшее число версии библиотеки в `/lib`.

Например, библиотека `libc` версия 4.4.1 имеет старшую цифру версии 4. Файл, содержащий библиотеку - `libc.so.4.4.1`. Символическая связь с именем `libc.so.4`, указывающая на этот файл, также в `/lib`. Вы должны изменить эту символическую связь, когда модифицируете библиотеки. Например, когда идет смена версий, вы должны изменить символическую связь файла `libc.so.4` на новую версию.

Важное замечание! Надо менять символическую связь за один шаг, как показано ниже. Если вы каким-то образом удалили символическую связь `libc.so.4` тогда программы, которые зависят от этой связи (включая базовые утилиты вроде `ls` и `cat`)

---

[37] (прим. переводчика: На момент перевода книги последняя версия была 5.3.9.)

перестанут работать. Используйте следующую команду для обновления символической связи `libc.so.4`, чтобы она указывала на файл `libc.so.4.4.1`:

```
# ln -sf /lib/libc.so.4.4.1 /lib/libc.so.4
```

Вы должны также изменить символическую связь `libm.so.`*version* таким же манером. Если вы переходите на отличную (от прежней) версию библиотек, замените имена вышеупомянутых файлов. Пояснения к версии библиотеки должны прояснить детали. (Дополнительную информацию про символические связи смотрите в Разделе 3.10).

4.6.3 Модернизация gcc

Компиляторы gcc C и C++ используются для компиляции программ вашей системы, в первую голову - ядра. Новейшую версию gcc можно найти на FTP-серверах Linux. На `sunsite.unc.edu` его можно найти в каталоге `/pub/Linux/GCC` (вместе с библиотеками). Должен существовать файл версии для дистрибуции gcc, детализирующий, какие файлы вы должны переписать и как их инсталлировать.

4.6.4 Модернизация других программ

Модернизация других программ, это в основном проблема добычи соответствующих файлов и их инсталляции. Большинство программ для Linux распространяются как зазипованные tar-файлы, включая как исходные, так и выполняемые, или те и другие. Если выполняемые файлы не включены в версию, вам может потребоваться самостоятельно их откомпилировать. Обычно это означает запуск make в каталоге, где находятся исходники.

Чтение группы новостей USENET `comp.os.linux.announce` - простейший путь, чтобы выловить информацию о новых программах. Так что самый простой способ отыскать какие-то программы, это побродить по FTP-серверам, поскачивать с серверов (`ls-lR`) индексные файлы и, используя `grep`, найти желаемые файлы. Если вам доступен архив, это также может помочь. Детали смотрите в Приложении A. (Если у вас нет архива, вы можете по `telnet` выйти на архивный сервер вроде `archie.rutgers.edu`, войти как "`archie`" и воспользоваться командой "`help`"). Детали смотрите в Приложении A.

Один из удобных источников программ Linux - дисковый образ Slackware. Каждый диск содержит ряд файлов `.tgz`, просто зазипованных `tar`-файлов. Вместо переписывания диска вы можете переписать желаемые `.tgz` файлы из каталогов Slackware на FTP-сервере и прямо их инсталлировать. Если вы используете дистрибутив Slackware, команда `setup` может автоматически загрузить и инсталлировать полный набор дисков.

И еще раз, обычно не самое умное дело заниматься модернизацией путем переинсталляции новейшей версии Slackware или другого дистрибутива. Если вы таким образом реинсталлируете, вы обязательно уничтожите ваш сегодняшний вариант, включая каталоги пользователей и ваши стандартные настройки. Лучше заниматься модернизацией по частям, то есть, если нашлась программа, которую вы часто используете и которая имеет новую версию, модернизируйте ее. А иначе не волнуйте себя по пустякам. Правило усталого ветерана: "Если само не ломается - не трогай". Если ваша система работает - нет достаточных оснований для модернизации.

4.7 Управление файловыми системами

Другая задача системного администратора - забота о файловой системе. Большая часть этой работы состоит в проверке файловой системы на наличие поврежденных или испорченных файлов; многие системы делают такие проверки во время загрузки.

4.7.1 Монтирование файловых систем

Сначала несколько концепций, связанных с файловыми системами. Прежде, чем файловая система будет принята вашей системой, она должна быть **примонтирована** к какому-то каталогу. Например, если у вас файловая система на дискете, то вы должны примонтировать ее в каталог, скажем `/mnt`, для того, чтобы обеспечить доступ к ее файлам (смотрите Раздел 4.6.2). После монтирования файловой системы все файлы этой системы появляются в этом каталоге (и ниже). После размонтирования файловой системы каталог (в нашем случае `/mnt`) будет пуст, то же самое справедливо для файловой системы на жестком диске.[38] Система автоматически монтирует файловые системы на ваш жесткий диск во время загрузки. Так называемая "корневая файловая система" монтируется к каталогу `/`. Если у вас отдельные файловые системы, например, для `/usr` - она монтируется на `/usr`. Если у вас только корневая файловая система, то все файлы, включая содержимое `/usr`, существуют в этой файловой системе.

Команда `mount` используется для монтирования файловой системы.

```
mount -av
```

Выполняется из файла `/etc/rc` (файла системной инициализации во время загрузки, смотрите Раздел 4.10.1). Команда `ount -av` получает информацию о файловых

---

[38] (Прим. переводчика: Каталог `/mnt` будет пуст, если он был пуст до монтирования, иначе наоборот, станут видными файлы каталога `/mnt` (основной системы), которые становятся "невидимыми", когда к этому каталогу монтируется файловая система).

системах и монтирует в соответствии с файлом `/etc/fstab`. Пример файла `fstab` показан ниже.

```
# device       directory      type     options
/dev/hda2       /             ext2     defaults
/dev/hda3       /usr          ext2     defaults
/dev/hda4       none          swap     sw
/proc           /proc         proc     none
```

Первое поле - это устройство (имя монтируемого раздела). Второе поле - точка монтирования. Третье поле - тип файловой системы (например, `ext2` для системы типа ext2fs или `minix` для Minix filesystems). Таблица 4.1 перечисляет различные типы файловых систем, доступных в Linux.

Эта таблица для ядра версии 1.1.37.

Файловая система	Имя типа	Комментарий
Second Extended Filesystem	ext2	Наиболее распространенная для Linux
Extended Filesystem	ext	Вытеснена системой ext2
Minix Filesystem	minix	Файловая система Minix; редко используется
Xia Filesystem	xia	Похожа на ext2; редко используется
UMSDOS Filesystem	umsdos	Для инсталяции Linux на разделы MS-DOS
MS-DOS Filesystem	msdos	Для доступа к файлам MS-DOS
/proc Filesystem	proc	Дает информацию о процессах для ps и т.п.
ISO 9660 Filesystem	iso9660	Используется большинством CD-ROM
Xenix Filesystem	xenix	Для доступа к файлам из Xenix.
System V Filesystem	sysv	Для доступа к файлам из System V вариант для x86.
Coherent Filesystem	coherent	Для доступа из Coherent
HPFS Filesystem	hpfs	Доступ только на чтение для разделов HPFS (DoubleSpace).

Таблица 4.1. Типы Файловых систем Linux

Не все эти типы могут быть доступны на вашей системе; ваше ядро должно иметь соответствующую откомпилированную поддержку. О компиляции ядра смотрите в Разделе 4.7.

Последнее поле файла `fstab` (options) этобылопередТаблицей4.1 содержит опции монтирования, обычно они устанавливаются в “`defaults`”.

Вы можете видеть, что разделы своппинга также включены в `/etc/fstab`. Они имеют каталог монтирования ”tt/none/”, и тип ”`swap`”. Команда `swapon -a` выполняемая из

`/etc/rc` используется для обеспечения своппинга на все устройства, перечисленные в `/etc/fstab`.

Файл `fstab` содержит одну специальную запись для файловой системы `/proc`. Как говорилось в Разделе 3.11.1, файловая система `/proc` используется для хранения информации о системных процессах, доступной памяти и т.п. Если `/proc` не примонтирован, такие команды, как ps не будут работать.

Внимание! Команда `mount` может использоваться только `root`. Это для обеспечения безопасности системы. Вам не захочется, чтобы монтирование и размонтирование файловых систем зависело от прихоти рядовых пользователей. Есть несколько программных пакетов, которые дают возможность тем самым рядовым пользователям монтировать и размонтировать файловые системы (особенно на дискетах) не затрагивая безопасности системы.

Команда `mount -av` фактически монтирует все файловые системы, кроме корневой файловой системы (в ранее приведенной таблице - `/dev/hda2`). Корневая файловая система автоматически монтируется ядром во время загрузки.

Вместо использования `mount -av` вы можете примонтировать файловую систему вручную. Команда

```
# mount -t ext2 /dev/hda3 /usr
```

эквивалентна монтированию файловой системы на `/dev/hda3` в примере `fstab`, рассмотренном ранее.

Вам никогда не следует монтировать и размонтировать файловые системы вручную. Команда `mount -av` в `/etc/rc` позаботится о монтировании файловых систем во время загрузки. Файловые системы автоматически размонтируются командами `shutdown` или `halt` перед выключением системы.

4.7.2 Проверка файловых систем

Бывает полезно почаще проверять вашу файловую систему на наличие поврежденных и испорченных файлов. Некоторые системы автоматически проверяют свои файловые системы во время загрузки (с помощью соответствующих команд из `/etc/rc`).

Для проверки файловых систем используются команды, зависящие от типа файловой системы. Для файловой системы `ext2fs` (самый широко используемый тип), такой командой служит `e2fsck`. Например, команда

```
# e2fsck -av /dev/hda2
```

проверит файловую систему `ext2fs` на `/dev/hda2` и автоматически исправит ошибки.

Обычно бывает полезно размонтировать файловую систему перед

ее проверкой. Например команда

```
# umount /dev/hda2
```

Размонтирует файловую систему на `/dev/hda2`, после чего вы можете ее проверить. Есть одно исключение, вы не можете размонтировать корневую файловую систему. Для того, чтобы проверить размонтировнную корневую файловую систему вам следует использовать специальную boot/root дискету (смотрите Раздел 4.11.1). Вы также не можете размонтировать файловую систему, если хотя бы один из ее файлов "занят" (“busy”), т.е. используется действующим процессом. Например, вы не можете размонтировать файловую систему, если хотя бы один из текущих рабочих каталогов пользователя находится на этой файловой системе. Вы получите сообщение “`Device busy`”, если вы попытаетесь размонтировать используемую файловую систему.

Другая файловая система использует различные формы команды `e2fsck`, такие как `efsck` и `xfsck`. На некоторых системах вы можете просто использовать команду `fsck`, которая определит тип файловой системы и выполнит соответствующую команду.

Внимание! Необходимо немедленно перезагрузить операционную систему после проверки монтированной файловой системы, если были внесены какие-то изменения в файловую систему. (Хотя в общем случае проверять неразмонтированную файловую систему). Например, если `e2fsck` сообщает, что она исправила хотя бы одну ошибку в файловой системе, вам следует немедленно выполнить `shutdown -r`, чтобы перезагрузить систему. Это позволить системе "ресинхронизировать" информацию о файловой системе, после модификации ее с помощью `e2fsck`.[39]

Файловая система `/proc` никогда не нуждается в проверках такого рода. `/proc` - это файловая система памяти, управляемая непосредственно ядром.

4.8 Использование файла своппинга

Вместо того, чтобы резервировать специальные разделы для области своппинга, вы можете использовать файл. Однако, чтобы это сделать, вы должны инсталлировать программы Linux и предварительно сделать все, что необхоимо для создания файлов своппинга.

---

[39] (прим. переводчика: То есть сново согласовать содержимое буферов памяти с соответствующими фрагментами файловой системы на диске).

Если у вас есть инсталлированная система Linux, вы можете использовать следующие команды для создания файла свопинга. Ниже мы собираемся создать файл свопинга размером в 8208 блоков (около 8 Мбайт).

```
# dd if=/dev/zero of=/swap bs=1024 count=8208
```

Эта команда создает файл свопинга. Замените “`count=`” размером файла свопинга в блоках.

```
# mkswap /swap 8208
```

Эта команда инициализирует `swap`-файл; вновь замените имя и размер своп-файла соответствующими значениями.

```
# /etc/sync
# swapon /swap
```

Теперь в свопинге будет задействован файл `/swap`, который мы создали, после синхронизации, которая гарантирует, что файл был записан на диск.

Главная неприятность, связанная с использованием `swap`-файлов, состоит в том, что доступ к ним происходит через файловую систему. Это означает, что блоки, составляющие swap-файл могут быть не смежними на диске. То есть скорость свопинга при использовании swap-файла ниже, чем при использовании swap-раздела, для которой блоки всегда смежны (последовательны) и запросы на ввод/вывод происходят прямо к устройству.

Другая проблема, связанная с использованием `swap`-файла, это возможность испортить информацию в файловой системе - при использовании больших файлов свопинга существует шанс, что вы попортите систему, если что-то происходит неправильно. Имея раздел свопинга отдельно от файловой системы вы страхуетесь от таких неприятностей.

Использование файла свопинга может быть очень полезным, если у вас есть временная потребность в дополнительном пространстве для свопинга. Например, если вы компилируете большую программу и хотите ускорить дело, вы можете временно создать файл свопинга и использовать его в дополнение к имеющейся области свопинга.

Для того, чтобы избавиться от файла свопинга, вначале используйте `swapoff`

```
# swapoff /swap
```

А теперь вы можете смело удалить файл.

```
# rm /swap
```

Помните, что каждый файл своппинга (или раздел) может быть размером до 16 Мбайт, но вы можете использовать до 8 файлов своппинга или разделов на своей системе.

4.9 Разношерстные задачи

Хотите верьте, хотите - нет, но существует ряд хозяйственных задач, входящих в функции системного администратора, которые не попадают ни в одну из основных категорий.

4.9.1 Файлы установки системы

При загрузке системы некоторые сценарии автоматически выполняются системой до входа в нее пользователей. Далее следует описание того, что в это безвременье происходит. Во время загрузки ядро запускает процесс `/etc/init`. `init` - это программа, которая читает свои настроечные файлы (`/etc/inittab`) и запускает другие процессы, базирующиеся на содержании этих файлов. Один из важных процессов запускается из `inittab` - это `/etc/getty` , он стартует для каждой виртуальной консоли. Процесс `getty` захватывает ВК (Виртуальную Консоль) и запускает на ней процесс `login`. Это позволяет вам входить на каждой ВК. Если `/etc/inittab` не содержит процессов `getty` для конкретной ВК, на эту ВК вы не войдете.

Другой процесс, выполняемый из `/etc/inittab` - это `/etc/rc`, главный системный файл инициализации[40] Этот файл представляет из себя shell-сценарий, который выполняет любые необходимые команды инициализации во время загрузки, такие например, как монтирование файловых систем (смотрите Раздел 4.8) и инициализации области своппинга.

Ваша система может также выполнять и другие сценарии, например `/etc/rc.local`. `/etc/rc.local` обычно содержит команды инициализации, специфичные для вашей системы, такие как установка хост-имени (смотрите следующий раздел). `rc.local` может запускаться из `/etc/rc` или прямо из `/etc/inittab`.

---

[40] (прим. переводчика: или главный файл инициализации системы - что тоже верно).

4.9.2 Установка хост-имени

В сетевой среде хост-имя используется для однозначной идентификации конкретной машины, в то время как отдельно стоящей машине хост-имя придает чувство собственного достоинства и шарма. Это, как дать имя вашей собаке: вы можете обращаться к собаке просто "The dog"[41] , но значительно интереснее приписать собаке имя, вроде Spot или Woofie (или Шарик и Бобик).

Хост-имя элементарно устанавливается командой `hostname`. Если вы в сети, ваше хост-имя должно быть полным хост-именем вашей машины, таким как `goober.norelco.com`. Если вы не в сети, вы можете выбрать произвольные имена для хоста и домена, например `loomer.vpizza.com`, `shoop.nowhere.edu` или `floof.org`.

При установке хост-имени оно должно быть занесено в файл `/etc/hosts`, который приписывает IP адрес каждому хосту. Даже если ваша машина не в сети, вам следует включить ваше хост-имя в `/etc/hosts`. Например, если вы не имеете выхода в сеть по TCP/IP и ваше хост-имя `floof.org`, просто включите следующую запись в `/etc/hosts`:

```
127.0.0.1       floof.org localhost
```

Это припишет ваше хост-имя `floof.org` к локальному IP-интерфейсу (loopback address) 127.0.0.1 (используемому, даже если вы не в сети). Синоним `localhost` также приписывается этому адресу.

Если вы подключены к сети по TCP/IP, ваши действительные IP адрес и хост-имя должны появиться в `/etc/hosts`. Например, если ваше хост-имя `goober.norelco.com`, и ваш IP адрес 128.253.154.32, добавьте следующую строку в `/etc/hosts`:

```
128.253.154.32      goober.norelco.com
```

Если вашего хост-имени не будет в `/etc/hosts`, вы не сможете его установить. Для установки хост-имени просто используйте команду `hostname`. Например, команда

```
# hostname -S goober.norelco.com
```

устанавливает хост-имя `goober.norelco.com`. Во многих случаях команда hostname выполняется из из одного из системных установочных файлов, таких как `/etc/rc` или `/etc/rc.local`. Отредактируйте эти два файла и измените находящуюся там команду `hostname`, установив хост-имя своей машины; после перезагрузки машины хост-имя будет иметь новое значение.

[41] (прим. переводчика: это просто "собака" (с определенным артиклем - поскольку конкретная)

4.10 Что делать при ЧП

В некоторых случаях администратор системы будет сталкиватся с проблемой выкарабкивания из абсолютной катастрофы, такой например, как забытие пароля `root` или крах файловой системы. Лучший совет - *без паники*! Все делают глупые ошибки - это лучший способ освоить системное администрирование, хотя и патологический.

Linux не является нестабильной[42] версией UNIX. Действительно, у меня было значительно меньше проблем с зависанием системы, чем с коммерческими версиями UNIX на многих платформах. Linux также выигрывает от большого расположения к нему крутых программистов, которые могут помочь выпутаться из сложной ситуации.

Первый шаг в исследовании любой проблемы - это попытаться справиться с ней самостоятельно. Потыкайтесь там-сям и посмотрите, что из этого будет получаться. Слишком много времени системные администраторы тратят на рассылку во все стороны отчаянных воплей о помощи, прежде, чем вникнуть в проблему. В большем числе случаев вы обнаружите, что вы сами легко можете решить проблему. А это уже ваш прямой путь в мэтры.

Очень редки случаи, когда после краха системы требуется переинсталляция. Многие начинающие пользователи случайно удаляют некоторые важные системные файлы и немедленно бегут за инсталляционным диском. Это не "Боже мой"! Прежде чем применять такие отвратительные меры, исследуйте проблему и попросите других помочь ее решить. В большинстве случаев вы можете восстановить систему с дискеты сопровождения (maintenance diskette).

4.10.1 Восстановление с использованием дискеты сопровождения

Одно назаменимое средство для администратора системы - это так называемый “boot/root disk” - дискета, которая может загрузить полный Linux, вне зависимости от вашего жесткого диска. Boot/root disks в действительности очень прост - вы создаете корневую файловую систему на дискете, помещая на нее все необходимые утилиты, инсталлируя на дискете LILO и загружаемое ядро. Другой способ, это использовать одну дискету для ядра и другую для корневой файловой системы. В любом случае результат одинаков: Вы запускаете Linux полностью с дискет.

Канонический пример boot/root disk - это загрузочный диск Slackware.[43] Эти дискеты содержат загрузочную таблицу и корневую файловую систему. Предполагается, что они

---

[42] (прим. переводчика: так в оригинале)

[43] (Смотрите Раздел 2.1.1 относительно информации по перекачке его по Internet. Для этого вам не надо скачивать полностью - только `boot` и `root` дискеты).

используются при инсталляции дистрибутивов Slackware, но бывают очень полезны для сопровождения системы.

boot/root disk, созданный H.J Lu, который можно взять в `/pub/Linux/GCC/rootdisk` на `sunsite.unc.edu` - другой пример такого рода диска сопровождения.

Или, если вы достаточно амбициозны, можете создать свой. Хотя, в большинстве случаев, использовать готовый boot/root disk - значительно легче и надежнее.

Использовать boot/root disk очень легко. Просто загрузите диск на вашей системе и войдите под `root` (обычно без пароля). Чтобы получить доступ к файлам вашего жесткого диска, необходимо примонтировать ваши файловые системы вручную. Например, команда

```
# mount -t ext2 /dev/hda2 /mnt
```

примонтирует файловую систему ext2fs на `/dev/hda2` под `/mnt`. Помните, что `/` теперь находится на boot/root disk; вам необходимо примонтировать файловую систему вашего жесткого диска под каким-то каталогом, чтобы получить доступ к файлам. Так что `/etc/passwd` вашего жесткого диска теперь в `/mnt/etc/passwd`, если вы примонтировали вашу корневую файловую систему на `/mnt`.

4.10.2 Восстановление пароля для root

Если вы забыли пароль вашего `root` - нет проблем. Просто загрузитесь с boot/root disk, примонтируйте вашу корневую файловую систему под `/mnt` и сотрите поле пароля для `root` в `/mnt/etc/passwd`, как например:

```
root::0:0:root:/:/bin/sh
```

Теперь `root` без пароля; когда вы перезагрузитесь с жесткого диска, вы сможете войти как `root` и снова установить пароль, используя команду `passwd`. Не правда ли, вы счастливы, что научились работать с `vi`? На вашей boot/root disk, редакторов, вроде Emacs наверняка нет, а `vi` должен быть.[44]

4.10.3 Восстановление файловой системы

Если у вас каким-то образом грохнулась файловая система, вы можете использовать `e2fsck` (это в случае, если вы используете файловую систему типа ext2fs) для

---

[44] (прим.переводчика: администратор должен отдавать себе отчет, что процедуру снятия пароля root умеет запросто выполнять не он один).

исправления попорченых данных файловой системы с дискет. Другие файловые системы используют другие формы команды `fsck`; детали смотрите в Разделе 4.8.

Когда вы проверяете вашу файловую систему с дискеты, лучше всего, чтобы файловая система не была примонтирована.

Частая причина неисправности файловой системы - порча суперблока. Суперблок, это "голова" (‘‘header’)’файловой системы, которая содержит информацию о статусе файловой системы, размере, свободных блоках и т.д. Если вы попортили ваш суперблок (например, случайно прямо в него записали какие-то данные) операционная система может вообще не распознать файловую систему. Все попытки примонтировать файловую систему потерпят неудачу, и `e2fsck` не поможет решить проблему.

К счастью, файловая система типа ext2fs сохраняет копии суперблока в границах "группы блоков" (‘‘block group’’) на диске, обычно через каждые 8K блоков. Для того, чтобы приказать `e2fsck` использовать копию суперблока, вы можете использовать команду

```
# e2fsck -b 8193 partition
```

где <`partition`> - это раздел, на которой располагается файловая система. Опция `-b 8193` приказывает `e2fsck` использовать копию суперблока, хранящуюся в блоке 8193 файловой системы.

4.10.4 Восстановление потерянных файлов

Если вы случайно удалили важные файлы, нет способа их "разудалить" обратно. Однако, вы можете скопировать соответствующие файлы с дискеты себе на жесткий диск. Например, если вы удалите `/bin/login` в своей системе (который обеспечивает вход в систему), просто загрузите boot/root дискету, примонтируйте корневую файловую систему на `/mnt` и используйте команду

```
# cp -a /bin/login /mnt/bin/login
```

Опция `-a` приказывает `cp` сохранить права доступа копируемых файлов. Разумеется, если удаленные файлы не столь существенны, что они не были удостоены копирования на дискету boot/root floppy, значит вам не повезло. Если вы создавали резервные копии, вы можете скопировать файлы оттуда.

4.10.5 Восстановление потерянных библиотек

Если вы случайно потеряли свои библиотеки или символические связи в `/lib`, скорее всего команды, которые зависят от этих библиотек, больше не будут выполняться (смотрите Раздел 4.7.2). Простейшее решение - загрузиться с дискеты boot/root, примонтировать вашу корневую файловую систему и восстановить библиотеки в `/mnt/lib`.

5 Дополнительные возможности

Эта глава познакомит вас с некоторыми из наиболее интересных возможностей Linux. Это предполагает, что вы имеете как минимум начальные знания UNIX и поняли информацию, содержащуюся в предыдущих разделах.

Наиболее важным аспектом Linux, который отличает его от других реализаций UNIX является его открытая концепция разработки.

Linux
не разрабатывался небольшой группой программистов, возглавляемой коммерческой структурой с целью получения прибыли.

Он разрабатывался постоянно растущей группой хакеров, вносящих то что им нравится в домашнее варево UNIX.

Linux включает в себя огромное количество разнообразного как по типу так и по способам разработки программного обеспечения. Некоторым не нравится недостаток единобразия, некоторые считают его основным преимуществом Linux.

5.1 X Window

X-Window является большой, мощной (и отчасти сложной) графической средой для UNIX систем. Система X-Window была разработана в Массачусетстком технологическом институте (MIT), которая стала затем стандартом для всех UNIX систем. Практически каждая рабочая станция UNIX в мире работает на одном из вариантов X-Window.

Группа программистов, возглавляемая Дэвидом Вексельблатом (David Wexelblat),[45] произвела перенос MIT X Window System версия 11, релиз 6 (X11R6) для 80386/80486/Pentium UNIX систем как свободно распространяемого программного

---

[45] (Вы можете связаться с Девидом по E-Mail: dwex@XFree86.org)

продукта. Эта версия, известная как XFree86 *TM*,[46] доступна для System V/386, 386BSD и других реализаций UNIX для процессоров x86, включая Linux. Она включает в себя все требуемые выполняемые коды, конфигурационные файлы, библиотеки и инструментарий.

Полная настройка и использование X Window выходит за пределы этой книги. Вам следует обратиться к книге : *The X Window System: A User's Guide* (см. Приложение A) В этой главе мы опишем шаг за шагом установку и настройку XFree86 для Linux. Для более детального ознакомления вы можете обратиться к документации, поставляемой вместе с XFree86 (она обсуждается ниже). Другим полезным источником информации является *THE LINUX XFree86 HOWTO*.

5.1.1 Требования к аппаратуре

XFree86 версии 3.1, вышедшая в сентябре 1994, года поддерживает следующие микросхемы видеоадаптеров. (Прежде чем устанавливать XFree86 вам надо выяснить тип микросхемы вашего видеоадаптера.)

Документация, поставляемая вместе с видеоадаптером, как правило указывает тип используемых микросхем. Если вы приобрели новую видеокарту или новый компьютер, попросите поставщика уточнить изготовителя, модель и тип микросхем видеокарты. Как правило поставщики охотно дадут вам эту информацию. Большинство из них сообщит, что видеокарта является стандартной SVGA картой и будет работать в вашей операционной системе. Об'ясните им, что ваше программное обеспечение (Linux и XFree86) не поддерживает всех видеокарт и вам требуется дополнительная информация.

Вы можете также определить тип микросхемы, вызвав команду `SuperProbe`, входящую в состав XFree86. Это будет описано ниже.

XFree86 версии 3.1, вышедшая в сентябре 1994 года поддерживает следующие типы микросхем:

- Tseng ET3000, ET4000AX, ET4000/W32
- Western Digital/Paradise PVGA1
- Western Digital WD90C00, WD90C10, WD90C11,WD90C24,WD90C30, WD90C31, WD90C33
- Genoa GVGA

---

[46] (XFree86 является торговой маркой XFree86 Project, Inc.)

- Trident TVGA8800CS, TVGA8900B, TVGA8900C, TVGA8900CL, TVGA9000, TVGA9000i, TVGA9100B, TVGA9200CX, TVGA9320, TVGA9400CX, TVGA9420
- ATI 18800, 18800-1, 28800-2, 28800-4, 28800-5, 28800-6, 68800-3, 68800-6, 68800AX, 68800LX, 88800
- NCR 77C22, 77C22E, 77C22E+
- Cirrus Logic CLGD5420, CLGD5422, CLGD5424, CLGD5426, CLGD5428, CLGD5429, CLGD5430, CLGD5434, CLGD6205, CLGD6215, CLGD6225, CLGD6235, CLGD6420
- Compaq AVGA
- OAK OTI067, OTI077
- Avance Logic AL2101
- MX MX68000, MX680010
- Video 7/Headland Technologies HT216-32

Поддерживаются также следующие адаптеры с графическими ускорителями:

- 8514/A (and true clones)
- ATI Mach8, Mach32
- Cirrus CLGD5420, CLGD5422, CLGD5424, CLGD5426, CLGD5428, CLGD5429, CLGD5430, CLGD5434, CLGD6205, CLGD6215, CLGD6225, CLGD6235
- S3 86C911, 86C924, 86C801, 86C805, 86C805i, 86C928, 86C864, 86C964
- Western Digital WD90C31, WD90C33
- Weitek P9000
- IIT AGX-014, AGX-015, AGX-016
- Tseng ET4000/W32, ET4000/W32i, ET4000/W32p

Видеокарты с этими микросхемами поддерживаются для всех системных шин, включая VLB и PCI.

Все вышеперечисленные карты поддерживаются как в режиме 256 цветов, так и в монохромном режиме, за исключением Avance Logic, MX and Video 7 микросхем, которые поддерживаются только в режиме 256 цветов. Если на вашей видеокарте установлено достаточно видеопамяти, многие из микросхем поддерживются в режиме

16 и 32 бита на точку (65 тысяч и 4 млн цветов) (некоторые из Msch32, P9000, S3 и Cirrus видеокарт). Обычно видеокарты используются в режиме 8 бит на точку (256 цветов).

Монохромный сервер поддерживает основные карты VGA, монохромные карты Hercules, Hyundai HGC1280, Sigma LaserView иу Apoll. На карте Compaq AVGA в монохромном режиме поддерживается только 64Кб видеопамяти и работа карты GVGA с памятью более 64Кб не проверена в настоящее время.

Этот список несомненно расширится со временем. Полный список поддерживаемых карт вы найдете в замечаниях к текущей версии XFree86.

Одной из проблем, с которой столкнулись разработчики, являлся нестандартный механизм определения частоты, используемый для управления картой. Некоторые производители либо не описывали способ программирования карты, либо требовали подписания дополнительного соглашения о нераспространении полученной информации. Это очевидно ограничило бы свободное распространение XFree86, чего естественно не могли допустить разработчики. Долгое время данная проблема была с видеокартами, производимыми фирмой Diamond, но начиная с версии 3.1 XFree86, Diamond начала сотрудничество с разработчиками с целью выпуска драйвера для этой карты.

Предполагаемая конфигурация компьютера для установки XFree86 под Linux включает в себя как минимум 8 мегабайт оперативной памяти и видеокарту с одной из вышеперечисленных микросхем. Для оптимальной работы мы советуем использовать видеокарту с графическим ускорителем, например S3.

Перед покупкой дорогостоящей видеокарты вам следует просмотреть документацию по XFree и убедиться, что выбранная вами карта поддерживается. Сравнительные тесты Benchmark для различных видеокарт под XFree86 периодически выставляются в конференции `comp.windows.x.i386unix` и `comp.os.linux.misc`.

Замечу, что мой персональный компьютер с Linux содержит 486DX2-66, 20 мегабайт RAM, и имеет VLB S3-864 видеоадаптер с 2 мегабайтами оперативной памяти. Я протестировал X benchmarks на этой машине и на рабочей станции Sun Sparc IPX. Linux где-то раз в 7 быстрее, чем Sparc IPX (для любопытных, XFree86-3.1 под Linux обеспечил скорость 171,000 xstones; Sparc IPX - около 24000). Обычно, XFree86 под Linux с графическим ускорителем показывает существенно большую производительность чем коммерческие рабочие станции (которые обычно используют неэффективные алгоритмы обработки графической информации).

Для вашей машины требуется как минимум 4 мегабайта оперативной памяти и 16 мегабайт виртуальной (например, 8 мегабайт оперативной памяти и 8 мегабайт своппинга). Имейте в виду, что чем больше физической оперативной памяти вы имеете, тем меньше операционная система использует своппинг. Так как операция своппинга исходно медленная (доступ к диску намного медленнее, чем к памяти), для комфортабельной работы вам следует иметь 8 или более мегабайт. Система с 4-мя мегабайтами работает намного (в десятки раз) медленнее чем с 8-ю мегабайтами.

Установка XFree86 Дистрибутив Xfree86 в выполняемых кодах можно найти на целом ряде FTP-серверов. На `sunsite.unc.edu` он находится в каталоге `/pub/Linux/X11`. (На момент написания текущая версия была 3.1; периодически появляются новые версии).

Вполне возможно, что вы имеете XFree86 как часть дистрибутива Linux, в этом случае в перекачке XFree86 нет необходимости.

Если же вы собираетесь скачать XFree с FTP-сервера, следующая таблица содержит список файлов в дистрибутиве XFree86-3.1.

Вам потребуется один из серверов:

Файл	Описание
XF86-3.1-8514.tar.gz	Сервер для 8514 видеокарт.
XF86-3.1-AGX.tar.gz	Сервер для AGX видеокарт.
XF86-3.1-Mach32.tar.gz	Сервер для Mach32 видеокарт.
XF86-3.1-Mach8.tar.gz	Сервер для Mach8 видеокарт.
XF86-3.1-Mono.tar.gz	Сервер для монохромного режима.
XF86-3.1-P9000.tar.gz	Сервер для P9000 видеокарт.
XF86-3.1-S3.tar.gz	Сервер для S3 видеокарт.
XF86-3.1-SVGA.tar.gz	Сервер для Super VGA видеокарт.
XF86-3.1-VGA16.tar.gz	Сервер для VGA/EGA видеокарт.
XF86-3.1-W32.tar.gz	Сервер для ET4000/W32 видеокарт.

все нижеперечисленные файлы:

Файл	Описание
XF86-3.1-bin.tar.gz	Остальные программы X11R6.
XF86-3.1-cfg.tar.gz	Конфигурационные файлы для xdm, xinit и fs.
XF86-3.1-doc.tar.gz	Документация и руководства.
XF86-3.1-inc.tar.gz	Include файлы. (?)
XF86-3.1-lib.tar.gz	Разделяемые библиотеки.
XF86-3.1-fnt.tar.gz	Основные фонты.

следующие файлы не являются обязательными:

Файл	Описание

XF86-3.1-ctrb.tar.gz	Выбранные дополнительные программы (?).
XF86-3.1-extra.tar.gz	Дополнительные сервера для XFree86.
XF86-3.1-lkit.tar.gz	Инструментарий для компиляции серверов.
XF86-3.1-fnt75.tar.gz	Фонты 75-dpi.
XF86-3.1-fnt100.tar.gz	Фонты 100-dpi.
XF86-3.1-fntbig.tar.gz	Large Kanji и другие фонты.
XF86-3.1-fntscl.tar.gz	Масштабируемые фонты (Speedo, Type1).
XF86-3.1-man.tar.gz	Руководства.
XF86-3.1-pex.tar.gz	Выполняемые файлы, include-файлы, библиотеки для PEX.
XF86-3.1-slib.tar.gz	Статические библиотеки.
XF86-3.1-usrbin.tar.gz	Программы-демоны, размещающиеся в /usr/bin.
XF86-3.1-xdmshdw.tar.gz	Версия программы xdm с поддержкой теневых паролей.

Каталог XFree должен содержать файлы `README` и замечания по инсталляции текущей версии.

Все что вам требуется для инсталляции XFree86, это получить указанные файлы, создать каталог `/usr/X11R6` (пользователем `root`), перейти в этот каталог и распаковать файлы. Например:

```
# gzip -dc XF86-3.1-bin.tar.gz | tar xfB -
```

Имейте в виду, что эти файлы упакованы относительно каталога `/usr/X11R6`, так что необходимо распаковывать их находясь в этом каталоге.

После распаковки файлов, вам необходимо связать файл `/usr/X11R6/bin/X` с тем сервером, который вы намереваетесь использовать. Например, если вы желаете работать с SVGA сервером, файл `/usr/bin/X11/X` необходимо связать с файлом `/usr/X11R6/bin/XF86_SVGA`. Если же вы собираетесь использовать монохромный сервер, переустановите связь командой:

```
# ln -sf /usr/X11R6/bin/XF86MONO  /usr/X11R6/bin/X
```

Это же справедливо и для серверов других видеокарт.

Если вы не уверены какой сервер использовать, или не знаете какую микросхему содержит ваша видеокарта, вы можете запустить команду `SuperProbe` (включенную в XF86-3.1-bin). Эта программа попытается определить тип микросхемы вашей видеокарты и другую полезную информацию.

Вам следует убедиться, что каталог `/usr/bin/X11R6` находится в переменной среды PATH. Это может быть сделано редактированием файлов /etc/profile или `/etc/csh.login` (в зависимости от оболочек (shell) которые вы или другие пользователи используют). Вы также можете просто добавить этот каталог с вашей переменной `PATH`, корректируя в вашем домашнем каталоге файлы `.bashrc` или `.cshrc`, в зависимости от типа вашей оболочки.

Вам также необходимо обеспечить загрузку динамических библиотек. Для этого добавьте строку:

```
/usr/X11R6/lib
```

в файл `/etc/ld.so.conf` и запустите команду `/sbin/ldconfig` как суперпользователь.

5.1.2 Настройка XFree86

В большинстве случаев установка XFree не представляет проблем. Однако, если вы желаете использовать видеокарту для которой драйвер находится в процессе разработки или добиться лучших разрешения или производительности от карты с графическим акселератором, то вам потребуется определенное время для настройки XFree.

В этой главе мы опишем как создать и отредактировать `XF86Config` файл, который настраивает XFree86 сервер.

В большинстве случаев лучше всего начать с ``основной'' XFree86 конфигурации, которая использует низкое разрешение , например 640x480, поддерживаемого всеми видеокартами и мониторами. Однажды настроив XFree на стандартное разрешение, вы можете затем подстроить файл конфигурации для для того, чтобы использовать все возможности, предоставляемые вашей аппаратурой.

В дополнение к информации приведенной здесь, вам следует ознакомиться со следующей документацией:

- Документация по XFree в каталоге `/usr/X11R6/lib/X11/doc` (включенного в пакет XFree86-3.1-doc). Вам следует особенно обратить внимане на файл `README.Config`.
- Многие видео микросхемы имеют отдельный справочный файл `README` (например `README.Cirrus`, `README.S3`). Прочитайте их, если вы имеете такую видеокарту.
- Руководства (man pages) для XFree86.
- Руководство (man) на `XF86Config`.
- Руководство (man) на конкретный, используемый вами сервер (например `XF86_SVGA` или `XF86_S3`).

Основным файлом настройки XFree86 является файл `/usr/X11R6/lib/X11/XF86Config`. Этот файл содержит информацию о вашей мыши, параметрах видеокарты и т п. В качестве примера дистрибутив XFree86 содержит файл `XF86Config.eg`. Скопируйте его в файл `XF86Config` и отредактируйте.

`XF86Config` man page подробно об'ясняет формат этого файла. Прочитайте данный документ , если вы еще это не сделали. Далее мы собираемся просмотреть файл `XF86Config` участок за участком. Этот файл может выглядеть не совсем так, как файл в вашем дистрибутиве XFree86, но структура их совпадает.

!Заметьте, что формат файла `XF86Config` может изменяться с каждой версией XFree86; эта информация может быть верной только для XFree86 версии 3.1.

!Имейте также в виду, что не следует просто копировать конфигурационный файл, приведенный здесь и пытаться использовать его. Попытка использовать конфигурационный файл, не соответствующий вашему оборудованию, может заставить ваш монитор работать со слишком высокой для него частотой; были сообщения о выходе из строя мониторов (особенно мониторов с фиксированной частотой) при использовании неверных `XF86Config` файлов.

Каждая секция файла `XF86Config` определяется парой строк Section "`<section-name>`" ... `EndSection`. Первая секция файла называется `Files`, и выглядит следующим образом:

```
Section "Files"
      RgbPath     "/usr/X11R6/lib/X11/rgb"
      FontPath    "/usr/X11R6/lib/X11/fonts/misc/"
      FontPath    "/usr/X11R6/lib/X11/fonts/75dpi/"
EndSection
```

Строка `RgbPath` указывает местоположение базы данных цветов, а каждая строка `FontPath` определяет каталог, содержащий фонты. Как правило, вам не следует изменять эти строки, вам следует только убедиться, что все каталоги фонтов присутствуют.

Следующая секция имеет имя `ServerFlags` и определяет несколько глобальных параметров для сервера. Как правило эта секция пуста.

```
Section "ServerFlags"
# Uncomment this to cause a core dump at the spot where a signal is
# received. This may leave the console in an unusable state, but may
# provide a better stack trace in the core dump to aid in debugging
#    NoTrapSignals
# ncomment this to disable the CrtlAltBS server abort sequence
#    DontZap
EndSection
```

Все строки данной секции закомментированы.

Следуюшая секция `Keyboard`. Она определяет работу клавиатуры.

```
Section ”Keyboard”
      Protocol    ”Standard”
      AutoRepeat  500 5
      ServerNumLock
EndSection
```

Доступны также и другие режимы. Описанные выше опции работают на большинстве клавиатур.

Следующая секция - `Pointer` определяет параметры мыши.

```
Section ”Pointer”

      Protocol    ”MouseSystems”
      Device      ”/dev/mouse”

# Baudrate and SampleRate are only for some Logitech mice
#    BaudRate   9600
#    SampleRate 150

# Emulate3Buttons is an option for 2-button Microsoft mice
#    Emulate3Buttons

# ChordMiddle is an option for some 3-button Logitech mice
#    ChordMiddle

EndSection
```

Единственными опциями, на которые стоит обратить внимание являются `Protocol` и `Device`. `Protocol` определяет протокол, который использует ваша мышь. Возможными типами (для Linux - есть другие опции, доступными для других ОС) являются:

- BusMouse
- Logitech
- Microsoft
- MMSeries
- Mouseman
- MouseSystems

- PS/2
- MMHitTab

Для Logitech busmouse следует использовать протокол BusMouse. Заметьте, что для старых мышей Logitech необходимо использовать протокол `Logitech`, а для новых или `Microsoft`, или `Mouseman` протокол.

Строка `Device` определяет устройство к которому подключена мышь. На большинстве систем Linux это `/dev/mouse`. `/dev/mouse` обычно связано с соответствующим серийным портом (например `/dev/cua0` - COM1 или `/dev/cua1` - COM2) или с портом `busmouse`. В любом случае убедитесь, что указанное устройство есть в каталоге `/dev` и работоспособно.

Следующая секция - `Monitor`, определяет характеристики вашего монитора. Файл `XF86Config` может содержать не одну, а несколько секций `Monitor` (это справедливо и для других секций). Это полезно в том случае, когда вы подключили к системе несколько мониторов или используете один и тот же `XF86Config` файл для различных конфигураций.

```
Section "Monitor"

  Identifier  "CTX 5468 NI"

  # These values are for a CTX 5468NI only! Don´t attempt to use
  # them with your monitor (unless you have this model)

  Bandwidth    60
  HorizSync    30-38,47-50
  VertRefresh  50-90

  # Modes:  Name   dotclock    horiz               vert

  ModeLine "640x480"   25      640 664 760 800     480 491 493 525

  ModeLine "800x600"   36      800 824 896 1024    600 601 603 625
  ModeLine "1024x768"  65      1024 1088 1200 1328 768 783 789 818

EndSection
```

Строка `Identifier` используется для именования текущего описания монитора. Это может быть любая строка, на которую вы в дальнейшем ссылаться в файле `XF86Config`.

`HorizSync` определяет возможные скорости горизонтальной развертки для вашего монитора в Килогерцах. Если у вас многочастотный (multisync) монитор, вы можете указать интервал значений (или несколько интервалов, разделенных запятой), как показано выше. Если у вас монитор с фиксированной частотой, то вам надо указать список фиксированных значений. Например:

```
HorizSync    31.5, 35.2, 37.9, 35.5, 48.95
```

В руководстве на ваш монитор эти значения должны быть описаны. Если вы не имеете этой информации, вам следует связаться с производителем или продавцом вашего монитора.

Строка `VertRefresh` описывает возможные значения частоты вертикальной развертки для вашего монитора в герцах. Как и для `HorizSync` вы можете указать интервал или список дискретных значений. Ваше руководство на монитор должно содержать эту информацию.

Сервер использует значения `HorizSync` и `VertRefresh` только для того, чтобы убедиться что вы верно определили разрешение монитора. Это исключает возможность разрушения монитора при попытке работы с ним на частоте превышающей максимально допустимую.

Строка `ModeLine` определяет один из режимов разрешения вашего монитора. Ее формат:

```
ModeLine name clock horiz-values vert-values
```

<`name`> - строка, которую вы можете использовать в этом файле в дальнейшем для указания разрешения. <`dot-clock`> определяет частоту задающего генератора адаптера для этого разрешения. Обычно частота указывается в мегагерцах. Она определяет скорость с которой видеокарта должна посылать значения точек экрана на монитор при указанном разрешении. <`horiz-values`> и <`vert-values`> состоят из 4-х цифр каждая. Эти значения определяют, когда электронная пушка монитора во время развертки должна включиться и когда должны проходить импульсы горизонтальной и вертикальной синхронизации во время развертки луча.

Как описать строку `ModeLine` для вашего монитора? Файл `VideoModes.doc`, включенный в дистрибутив XFree86, детально описывает как определить эти значения для каждого разрешения, которое поддерживает ваш монитор. Значение `clock` должно соответствовать частотам, которые поддерживает ваша видеокарта. Далее в файле `XF86Config` вы определите эти значения.

Существует два файла `modeDB.txt` и `Monitors` в дистрибутиве XFree, которые могут содержать данные `ModeLine` для вашего монитора. Эти файлы располагаются в каталоге `/usr/X11R6/lib/X11/doc`.

Вы можете начать со значений `ModeLine` для мониторов стандарта VESA. Этот режим поддерживается большинством мониторов. Файл `modeDB.txt` включает описания синхронизации для стандартного разрешения VESA. В этом файле вы найдете такие строки:

```
# 640x480@60Hz Non-Interlaced mode
# Horizontal Sync = 31.5kHz
# Timing: H=(0.95us, 3.81us, 1.59us), V=(0.35ms, 0.064ms, 1.02ms)
#
# name      clock   horizontal timing    vertical timing    flags
"640x480"   25.175  640  664  760  800   480  491  493  525
```

Это стандартная строка синхронизации для разрешения 640x480 точек. Она устанавливает частоту 25.175 Mhz, которая должна поддерживаться большинством мониторов (более подробно об этом позже). В вашем файле эта строка должна выглядеть так:

```
ModeLine "640x480" 25.175  640 664 760 800  480 491 493 525
```

Заметим, что аргумент name в строке `ModeLine` (в нашем случае "640x480") может быть любой строкой, которая описывает для вас разрешение монитора.

Для каждой строки `ModeLine` сервер проверяет, попадают ли указанные значения в интервал указанных значений `Bandwidth`, `HorizSync` и `VertRefresh`. Если нет, то сервер выдаст предупреждение при начале работы.

Если стандартные значения синхронизации VESA не работают у вас, то просмотрите другие значения в файлах `modeDB.txt` и `Monitors` для других типов мониторов. Заметим, что многие 14 и 15 дюймовые мониторы не могут поддерживать разрешений 1024x768 точек из-за низкого значения `Bandwidth`. То есть, если вы не нашли описание режима высокого разрешения для вашего монитора, то не исключено, что ваш монитор не поддерживает такое разрешение.

Если вы потерпели неудачу с подбором строки ModeLine, изучите инструкцию описанную в файле `VideoModes.doc` вашего дистрибутива. По этой инструкции вы сможете описать строку `ModeLine` по спецификациям, описанным в вашем руководстве на монитор.

В конце концов, если вы не можете подобрать правильные значения строки `ModeLine`, то вы можете просто слегка изменить эти значения для достижения требуемого результата. Например, если изображение на экране слегка уходит влево или вверх, вы можете по инструкции из файла `VideoModes.doc` настроить значения синхронизации. Проверьте также наличие управляющих клавиш на вашем мониторе! Частенько бывает достаточным изменить горизонтальный и вертикальный размер изображения во время работы XFree чтобы добиться желаемой центровки и размера изображения. Наличие этих клавиш на монитора значительно упрощает жизнь.

Следующая секция `Device` описывает параметры вашей видеокарты. Например:

```
Section "Device"
          Identifier "#9 GXE 64"

          # Nothing yet; we fill in these values later.

EndSection
```

Эта секция описывает возможности вашей карты. `Identifier` определяет имя этого описания для ссылки на него в дальнейшем.

Первоначально вам не стоит заполнять эту секцию, за исключением поля `Identifier`. X-сервер можно использовать в режиме определения параметров установленной видеокарты. После определения этих параметров вы занесете их в эту секцию. X-сервер способен определить тип микросхемы видеокарты, поддерживаемый интервал частот, наличие RAMDAC и размер установленной памяти на видеоадаптере.

Прежде чем мы это сделаем, нам следует закончить описание файла `XF86Config`.

Следующая секция - `Screen` описывает возможные режимы работы X-сервера с видеокартой и монитором.

```
      Section "Screen"
          Driver     "Accel"
          Device     "#9 GXE 64"
          Monitor    "CTX 5468 NI"
          Subsection "Display"
                Depth      16
                Modes      "1024x768" "800x600" "640x480"
           ViewPort   0 0
           Virtual    1024 768
       EndSubsection
  EndSection
```

Строка `Driver` определяет тип сервера, который вы будете использовать. Вы можете использовать следующие сервера:

Accel:

Для XF86_S3, XF86_Mach32, XF86_Mach8, XF86_8514, XF86_P9000, XF86_AGX, and XF86_W32 серверов;

SVGA:

Для XF86_SVGA сервера;

VGA16

: Для XF86_VGA16 сервера;

VGA2

: Для XF86_Mono сервера;

Mono

: Для non-VGA монохромного драйвера в XF86_Mono и XF86_VGA16 серверах.

Убедитесь, что файл `/usr/X11R6/bin/X` является символьной ссылкой на используемый вами сервер.

Строка Device определяет идентификатор секции `Device`, описывающей установленную видеокарту. Выше мы описали секцию `Device` с идентификатором:

```
Identifier "#9 GXE 64"
```

Следовательно, здесь мы используем `"#9 GXE 64"` в строке `Device`.

Точно так же строка `Monitor` определяет имя секции `Monitor` для данного сервера, в данном примере `"CTX 5468 NI"`.

Подсекция Display определяет режим работы сервера при выводе информации на экран. Файл `XF86Config` детально описывает эти режимы. Режимы, которые вам необходимо знать:

- `Depth`. Опеделяет число битов на точку. Обычно `Depth` принимает значение 8 (256 цветов). Для сервера VGA16 вам следует установить значение `Depth` 4 и для монохромного сервера - 1. Если вы используете видеокарту с ускорителем и имеете достаточно памяти для поддержки большего числа бит на точку, Вы можете установить `Depth` 16 , 24 или 32. Если с этими значения появились проблемы вернитесь к значению 8 и попытайтесь решить проблему позже.
- `Modes`. Указывает список видеорежимов, описанных в секции ModeLines. Выше мы описали режимы Modelines названные `"1024x768"`, `"800x600"` и `"640x480"`. Следовательно строка Modes будет выглядеть:

```
Modes    "1024x768" "800x600" "640x480"
```

Первый режим, перечисленный в этой строке устанавливается по умолчанию после начала работы сервера. Далее вы можете переключать режимы разрешения изображения, использую клавиши *ctrl-alt-numeric+* и *ctrl-alt-numeric−*.

Лучше всего при начальном конфигурации XFree86 использовать минимальное разрешение, например 640x480, которое работает на большинстве систем. И после настройки этого режима настроить `XF86Config` на работу с большими разрешениями.

- `Virtual`. Устанавливает виртуальный размер экрана. XFree86 имеет возможность использовать дополнительнуб память на вашей видеокарте для расширения вашего рабочего поля.

  Когда указатель мыши доходит до края экрана, ваше рабочее поле сдвигается показывая новые части вашего рабочего поля. Следовательно, даже если вы работаете на мониторе с низким разрешением (например 800x600 точек), вы можете установить размер виртуального экрана насколько вам позволяет память видеоплаты (1 Мегабайтная плата может хранить рабочее поле 1024x768 с 256 цветами, 2-х Мегабайтная плата - 1280x1024 с 256 цветами или 1024x768 с 16384 цветами и т д). Конечно, вы не сможете увидеть сразу все поле на вашем мониторе, но вы можете легко просмотреть любую его часть.

  `Virtual` предоставляет вам прекрасную возможность использовать всю память вашего адаптера, но она довольно ограничена. Если вы желаете еще расширить возможности работы с экраном, вам следует использовать `fvwm`, `openwin` или другой подобный менеджер окон. `fvwm` и `openwin` позволяет вам иметь намного больший виртуальный экран (используя механизм спрятанных окон, вместо сохранения всего экрана в видеопамяти). Ваше виртуальное рабочее поле может состоять из 16x16 реальных экранов и более. Обратитесь к руководству по указанным командам. Большинство дистрибутивов XFree используют по умолчанию менеджер окон `fvwm`.

- `ViewPort`. Если вы использовали опцию `Virtual`, описанную выше, ViewPort устанавливает координаты левого верхнего угла виртуального экрана после начала работы сервера. Часто используют значение `Virtual 0 0`. Если вы не установили этого значения сервер центрирует виртуальный экран на мониторе (что может быть не всегда желательно).

Существуют и другие опции для данной секции (см. руководство для файла `XF86Config`). На практике же другие опции не обязательны для начальной установки сервера.

5.1.3 Заполнение информации о видеокарте

Теперь ваш файл `XF86Config` готов к использованию. Единственное, что мы не сделали - не заполнили информацию о видеокарте. Сейчас нам следует запустить X сервер в режиме определения видеокарты и дооформить `XF86Config` файл.

Эту информацию вы можете найти и в файлах `modeDB.txt`, `AccelCards` и `Devices` (все эти файлы находятся в каталоге `/usr/X11R6/lib/X11/doc`). Кроме этого существуют различные файлы `README` для конкретных микросхем. Вам следует просмотреть эти файлы и используя эту информацию (частоты, тип микросхем и другие режимы) доопределить файл `XF86Config`. Если какой то информации не хватает, вы можете определить ее путем описанным ниже.

В этом примере мы опишем настройку видеокарты `#9 GXE 64`, использующую микросхему S3. Эта карта одна из тех, с которыми работает автор, но все описанное ниже справедливо и для другой видеокарты.

Перво-наперво вам надо определить тип микросхемы, используемой видеокартой. Команда `SuperProbe` (располагающаяся в каталоге /usr/X11R6/bin) сообщит вам эту информацию, но вам необходимо знать под каким именем известна данная микросхема X серверу.

Чтобы определить это запустите команду:

```
X -showconfig
```

Сервер сообщит вам имена микросхем, с которыми он работает (руководство на X сервер также содержит эту информацию). Например, сервер XF86_S3 сообщит:

```
XFree86 Version 3.1 / X Window System
(protocol Version 11, revision 0, vendor release 6000)
Operating System: Linux
Configured drivers:
  S3: accelerated server for S3 graphics adaptors (Patchlevel 0)
        mmio928, s3generic
```

То есть сервер работает с микросхемами `mmio_928` и `s3_generic`. Руководство на сервер XF86_S3 описывает эти микросхемы и видеокарты, использующие их. В нашем случае видеокарта `#9 GXE 64` использует микросхему `mmio_928`.

Если вы не знаете какая микросхема стоит на видеокарте, X сервер может это определить. Запустите:

```
X -probeonly  /tmp/x.out 2&1
```

если вы работаете в оболочке shell. Если вы используете csh запустите:

```
X -probeonly & /tmp/x.out
```

Эту команду следует запускать при низкой загрузке компьютера. Эта команда определяет также частоту видеоадаптера и большая загрузка системы может исказить эти данные.

Выходная информация в файле `/tmp/x.out` будет содержать следующие строки:

```
XFree86 Version 3.1 / X Window System
(protocol Version 11, revision 0, vendor release 6000)
Operating System: Linux
Configured drivers:
  S3: accelerated server for S3 graphics adaptors (Patch level 0)
        mmio928, s3generic
Several lines deleted...
(--) S3: card type: 386/486 localbus
(--) S3: chipset:   864 rev. 0
(--) S3: chipset driver: mmio928
```

Мы видим, что сервер (XF86_S3) может работать с микросхемами `mmio_928` и `s3_generic`. Сервер протестировал видеокарту и опознал микросхему `mmio_928`. Следовательно, в секцию Device вам следует добавить строку, содержащую имя микросхемы, найденное сервером.

```
Section "Device"
          # We already had Identifier here...
          Identifier "#9 GXE 64"
          # Add this line:
          Chipset "mmio928"
EndSection
```

Теперь нам требуется определить частоты, поддерживаемые видеокартой. Как мы уже видели, каждый режим разрешения на мониторе требует определенной передачи точек от видеокарты. Нам необходимо определить какие частоты может обеспечить видеокарта.

Сначала следует просмотреть справочные файлы (`modeDB.txt`, и т п) описанные выше и определить, нет ли там описания частот вашей карты. Частоты, как правило представлены списком из 8 или 16-ти значений частот в Мегагерцах. Например в файле `modeDB.txt` можно найти строку описания видеокарты Cardinal ET4000:

```
chip     ram   virtual   clocks                        default-mode  flags
ET4000   1024  1024 768   25  28  38  36  40  45  32   0  "1024x768"
```

Как вы видите, данная карта поддерживает частоты: 25, 28, 38, 36, 40, 45, 32, and 0 MHz.

В секции `Device` файла `XF86Config`, вам следует добавить строку `Clocks` со списком частот. В нашем случае мы добавляем строку:

```
Clocks 25 28 38 36 40 45 32 0
```

к секции `Device`, после описания `Chipset`. Заметьте, что порядок частот важен! Вам не следует дублировать или изменять порядок частот.

Если вы не можете найти список частот для вашей карты, X сервер может также определить и эти значения. После вызова команды `X -probeonly`, описанного выше, вы увидите строку :

```
(--
) S3: clocks:  25.18  28.32  38.02  36.15  40.33  45.32  32.00  00.00
```

Теперь вам осталось лишь добавить строку `Clocks`, перечислив указанные значения. Так как часто список содержит 8 и более значений и не помещается в одной строке, вы можете продолжить список в следующих строках, только не забывайте сохранять порядок указанных сначений.

Перед запуском `X -probeonly`, уделитесь что в секции `Devices` нет строк описания `Clocks` или они закомментированы. Если эти значения уже есть, X сервер не будет проверять поддерживаемые платой частоты, а возьмет указанные в строке `Clocks`.

Заметьте, что некоторые видеокарты с акселератором используют микросхему с программируемой частотой (Смотрите руководство XF86_Accel; это в основном относится к картам S3, AGX и XGA-2 boards.) Эти микросхемы позволяют X-серверу сообщать карте какую использовать частоту. В этом случае мы вполне вероятно не сможем найти в вышеперечисленных файлах список частот для карты. Или список частот, выдаваемых командой `X -probeonly` будет содержать одно два значения с остальными дублированными или нулевыми значениями.

Для видеоплат, использующих микросхему программирования частоты, вам вместо строки `Clocks` следует использовать строку `ClockChip`. Эта строка задает имя микросхемы программирования частоты, установленной на карте. Руководства для каждого сервера описывает их имена. Например, в файле `README.S3` мы определили, что несколько S3-864 видеокарт используют микросхему “`ICD2061A`”. Следовательно, нам следует использовать строку:

```
ClockChip "icd2061a"
```

вместо строки `Clocks`. Так же как и строка `Clocks`, строка `ClockChip` должна быть в секции `Devices` после строки `Chipset`.

Некоторые карты с акселератором требуют определения в файле `XF86Config` строки `Ramdac`, описывающей тип используемой микросхемы RAMDAC. Руководство на сервер `XF86_Accel` описывает подробно опции этой строки. Как правило, X сервер верно определяет тип используемой микросхемы RAMDAC.

Некоторые видеокарты требуют определения нескольких дополнительных опций в секции `Devices`. Эти опции описаны как в руководствах на ваш X сервер, так и в справочных файлах (например `README.cirrus` или `README.S3`). Эти опции устанавливаются строкой `Options`. Например, видеокарта `#9 GXE 64` требует установку двух опций:

```
Option "numbernine"
Option "dac8bit"
```

Обычно X сервер работает и без этих опций, но с ними X сервер обеспечивает большую производительность. Существует слишком много всевозможных опций, чтобы из все здесь перечислить. Эти опции зависят от типа установленной видеокарты. Если вы вынуждены использовать эти опции - не волнуйтесь, руководства на X сервера и справочные файлы в каталоге `/usr/X11R6/lib/X11/doc/` об'яснят вам что они значат.

Итак, когда вы закончите, не забудьте завершить строкой `EndSection` секцию `Device`, которая будет выглядеть следующим образом:

```
Section "Device"
          # Device section for the #9 GXE 64 only !
          Identifier "#9 GXE 64"
          Chipset "mmio928"
          ClockChip "icd2061a"
          Option "numbernine"
          Option "dac8bit"
EndSection
```

Как уже сказано выше, большинство видеокарт требуют строку `Clocks` вместо строки `ClockChip`. Вышеприведенный пример применим только к конкретной видеокарте `#9 GXE 64`.

5.1.4 Запуск X-Windows

Как только вы закончите описание файла `XF86Config`, вы готовы запустить X сервер и начать работу. Сначала убедитесь, что каталог `/usr/X11R6/bin` включен в ваш путь (переменную `PATH`).

Для запуска X Window наберите команду:

```
startx
```

Это "оболочка" для команды `xinit` (если вы использовали `xinit` в других UNIX-системах).

Эта команда запускает X сервер и выполняет команды, найденные в файле `.xinitrc` в вашем домашнем каталоге. Если данного файла не существует, используется системный файл `/usr/X11R6/lib/X11/xinit/xinitrc`.

Стандартный `xinitrc` файл выглядит подобным образом:

```
#!/bin/sh

xterm -fn 7x13bold -geometry 80x32+10+50 &
xterm -fn 9x15bold -geometry 80x34+30-10 &
oclock -geometry 70x70-7+7 &
xsetroot -solid midnightblue &
exec twm
```

Этот расчет запускает два клиента `xterm` (эмулятор терминала), `oclock` (часы) и устанавливает темно-синий цвет экрана. Затем он запускает `twm` - оконный менеджер. Заметьте, что twm запускается через оператор exec. Оболочка /bin/sh, выполняющая этот расчет замещается командой `twm` и при окончании работы процесса `twm`, X-сервер также завершает свою работу. Вы можете выйти из `twm`, используя основное меню. Нажмите левую кнопку мыши, находясь на свободном месте экрана. На экране появится меню, которое позволит вам за выйти из `twm`, выбрав пункт `Exit Twm`.

Убедитесь, что последняя команда в файле `.xinitrc` запускается через exec и не запускается в фоне (нет символа `&` в конце строки). Иначе X сервер завершит свою работу, как только он запустит клиента из файла `.xinitrc`.

Кроме этого, вы можете выйти из X-а, нажав клавиши *ctrl-alt-backspace* одновременно.

Описанная выше конфигурация файла `.xinitrc` является очень простой. Если вы с ним немного поработаете вы можете получить множество отличных программ и конфигураций окон на экране. Например, оконный менеджер fvwm поддерживает виртуальные экраны, вы можете подобрать различные фонты, цвета, размеры окон, их позиции и так далее, все что вы пожелаете. Хотя система X Window может на первый взляд показаться простой, она чрезвычайно мощна и богата различными возможностями.

Если вы новичок в среде X Window, мы настоятельно рекомендуем вам приобрести книгу типа *The X Window System: A User's Guide*. Использование и настройка X-а довольно большая задача для того, чтобы описать ее в этой книге. В качестве дальнейших шагов посмотрите руководства для команд xterm, oclock, twm и т п.

5.1.5 Проблемы

Частенько случается, что у вас что-то не получается. Как правило, это связано с ошибками описания вашего файла `XF86Config`. Обычно, неверно указывают временные интервалы синхронизации монитора или частоты видеоплаты. Если у вас изображение на экране сдвинуто или его границы размыты, это точный показатель, что эти значения установлены неверно. Проверьте также, верно ли определили тип микросхемы видеокарты и другие опции в секции Device файла `XF86Config`. Убедитесь также, что вы используете необходимый X сервер и что файл `/usr/X11R6/bin/X` является символьной ссылкой на этот сервер.

Если это не поможет, попробуйте запустить X напрямую, используя команду:

```
X  /tmp/x.out 2&1
```

Затем остановите X сервер (нажав одновременно клавиши *ctrl-alt-backspace*) и проверьте содержимое файла `/tmp/x.out`. X сервер сообщит все предупреждения и ошибки, например о том, что ваша видеокартра на поддерживает необходимую для вашего монитора частоту.

Файл `VideoModes.doc`, включенный в дистрибутив XFree, содержит много советов по настройке вашего файла `XF86Config`.

Не забудьте, что вы можете использовать комбинации клавиш *ctrl-alt-numeric*+ и *ctrl-alt-numeric*− для переключения режимов разрешения монитора, перечисленых в секции Screen файла `XF86Config`. Если режим с высоким разрешением не работает, попытайтесь установить на меньшее разрешение. Это, по крайней мере, поможет определить вам ошибочные и правильные настройки вашего конфигурационного файла.

Попытайтесь также аппаратно подстроить ваш монитор, используя клавиши управления на мониторе.

Для обсуждения вопросов по XFree86 предназначены группа `comp.windows.x.i386unix` USENET. Неплохая идея - подписаться на эту конференцию и описать интересующие вас проблемы - может быть кто-то имеет такие же проблемы.

5.2 Доступ к файлам MS-DOS

Если, по какой-нибудь необ'яснимой прихоти, вам необходимо обеспечить доступ к файлам MS-DOS, вы можете это легко сделать.

Обычно для получения доступа к файлам MSDOS, вам достаточно примонтировать MS-DOS раздел или дискету и обращаться к фалам через файловую систему Linux. Например, если вы вставите дискету MS-DOS в устройство `/dev/fd0` (A: в нотации MS/DOS), команда

```
# mount -t msdos /dev/fd0 /mnt
```

примонтирует эту дискету к каталогу `/mnt`. Просмотрите секцию 4.6.2 для получения дополнительной информации о монтировании флоппи-дисков.

Точно также, вы можете примонтировать MS-DOS раздел на вашем винчестере. Если вы, например, имеете MS-DOS раздел на `/dev/hda1`, команда

```
# mount -t msdos /dev/hda1 /mnt
```

примонтирует ее. Не забудьте размонтировать DOS-раздел после окончания работы с ней. Вы можете монтировать раздел MS-DOS автоматически во время загрузки системы, если добавите строку в файл `/etc/fstab` (см. секцию 4.8). Например, следующая строка в файле `/etc/fstab` монтирует DOS раздел `/dev/hda1` на каталог `/dos`.

```
/dev/hda1     /dos     msdos      defaults
```

Вы можете также получить доступ к файлам MS-DOS, используя пакет Mtools. Команды `mcd`, `mdir` и `mcopy` этого пакеты работают точно также как команды MS-DOS `cd`, `dir`, `copy`. Если вы установили пакет Mtools, то он должен содержать и руководства на эти команды.

Доступ к файлам MS-DOS и выполнение программ MS-DOS - это две большие разницы. В настоящее время в процессе разработки находится эмулятор программ MS-DOS. Он широко распространен и даже входит в состав дистрибутива SLS. Доступен он также и по FTP с многих серверов (см. приложение C). Эмулятор MS-DOS достаточно полон для выполнения большинства DOS программ, включая Wordperfect. Однако Linux и MS-DOS совершенно разные операционые системы и полнота любого MS-DOS эмулятора в любой UNIX-системе всегда ограничена.

Кроме этого, в настоящее время разрабатывается в среде X Window эмулятор Microsoft Windows. Для получения дополнительной информации обратитесь к соответствующим группам новостей и FTP серверам.

5.3 Сетевая работа по протоколу TCP/IP

Linux поддерживает полный набор сетевых протоколов TCP/IP (Transport Control Protocol/Internet Protocol). TCP/IP стал наиболее успешно используемым механизмом работы в компьютерных сетях всего мира. С помощью Linux и карт Ethernet вы можете связать в локальную сеть ваши машины или (при соответствующем подключении) к Internet - всемирной сети TCP/IP.

Сцепить несколько UNIX-машин в небольшую локальную сеть (LAN) просто. Для этого требуется контроллер Ethernet в каждой машине, соответствующие кабели и еще некоторое сопутствующее оборудование. Или, если ваша фирма или университет имеют выход в Internet, вы можете просто к этой сети подцепиться со своей Linux-машиной.

Текущая реализация TCP/IP и соответствующие протоколы для Linux называются “NET-2”. Это не имеет отношения к так называемому релизу NET-2 для BSD UNIX. В данном контексте “NET-2” означает вторую реализацию TCP/IP для Linux.

Linux NET-2 также поддерживает протокол SLIP (Serial Line Internet Protocol). SLIP позволяет вам получить вход в Internet с помощью модема. Если ваша фирма или университет имеет выход по SLIP, вы можете выйти на SLIP-сервер и войти со своей машины в Internet по телефонной линии. И наоборот, если ваша Linux-машина имеет подключение по Ethernet к Internet, ваш Linux может исполнять функции SLIP-сервера.

Для получения более полной информации по установке TCP/IP под Linux, мы настоятельно советуем прочитать *Linux NET-2 HOWTO*, которое можно получить через FTP с `sunsite.unc.edu`. *NET-2 HOWTO* - это полное руководство по конфигурированию TCP/IP, включая связи по Ethernet и SLIP под Linux. `The Linux Ethernet HOWTO` описывает конфигурирование (настройку) различных драйверов карт Ethernet для Linux. Можно также воспользоваться `The Linux Network Administrator´s Guide` из проекта по документированию Linux - LDP (Linux Documentation Project). Более подробно про эти документы смотрите в Приложении A.

Интересна также книга: Craig Hunt `TCPIP` Network Administration/. Она содержит исчерпывающую информацию по использованию и настройке TCP/IP для систем UNIX.

5.3.1 Требования к аппаратуре

Вы можете использовать в Linux TCP/IP без какого-то дополнительного оборудования режим “loopback”, позволяющий разговаривать с самим собой. Это необходимо для ряда приложений и игр, использующих механизм “loopback”.

Но если вы хотите использовать Linux с сетевой работой через Ethernet по TCP/IP

, вы должны иметь одну из следующих карт Ethernet: 3com 3c503, 3c503/16; Novell NE1000, NE2000; Western Digital WD8003, WD8013; Hewlett Packard HP27245, HP27247, HP27250.

Имеется информация, что и следующие клоны работают: WD-80x3 clones: LAN-NET LEC-45; NE2000 clones: Alta Combo, Artisoft LANtastic AE-2, Asante Etherpak 2001/2003, D-Link Ethernet II, LTC E-NET/16 P/N 8300-200-002, Network Solutions HE-203, SVEC 4 Dimension Ethernet, 4-Dimension FD0490 EtherBoard 16, and D-Link DE-600, SMC Elite 16.

Дополнительную информацию по совместимости аппаратуры Ethernet в Linux можно найти в "Linux Ethernet HOWTO".

Linux также поддерживает SLIP, который позволяет использовать модем для выхода в Internet по телефонной линии. В этом случае вам нужен модем, совместимый с вашим SLIP-сервером - большинство серверов требует модемы на 14.4bps, V.32bis.[47]

5.3.2 Настройка TCP/IP на вашей системе

В этом разделе мы обсудим, как настраивать связь Ethernet - TCP/IP на вашей системе. Имейте в виду, что описываемый метод (предполагается, что) работает на многих системах, но, разумеется, не на всех. Этого обсуждения должно быть достаточно, чтобы указать вам правильный путь в настройке параметров сети на вашей машине. Но существует множество заковык и милых деталей, которые мы здесь даже не упоминаем. Мы, все-таки ориентируем вас на *Linux Network Administrators' Guide* и NET-2-HOWTO.

Прежде всего, мы предполагаем, что у вас есть Linux с инсталлированным TCP/IP. Это включает основных клиентов, таких как `telnet` и `ftp`, команды системного администратора, такие как `ifconfig` и `route` (обычно находящиеся в `/etc`), и сетевые настроечные файлы (такие как `/etc/hosts`). Другие, относящиеся к Linux сетевые документы, указанные выше, рассказывают, как инсталлировать сетевые программы Linux, если это еще не было сделано.

Мы также предполагаем, что ваше ядро было настроено и скомпилировано с поддержкой TCP/IP. Смотрите Раздел 4.7. по поводу компиляции ядра.

Когда это сделано, вы должны модифицировать ряд настроечных файлов, используемых NET-2. Для большинства это простая процедура. К сожалению, существует большое различие между дистрибутивами Linux относительно того, где

---

[47] (прим. переводчика: прогресс в модемах быстрее, чем в компьютерах, поэтому данные стареют еще быстрее; сейчас чаще можно услышать про 28.8bps и V.34, что тоже быстро устаревает).

должны размещаться различные конфигурационные файлы TCP/IP и поддерживающие программы. Чаще они могут быть обнаружены в `/etc`, но в других случаях их можно отыскать в `/usr/etc`, `/usr/etc/inet`, `/sbin` или в других неожиданных местах.

В худшем случае вы будете вынуждены использовать команду `find` для определения их местоположения в вашей системе. Имейте также в виду, что не все дистрибутивы хранят программы и файлы описания для NET-2 в одном месте - они могут быть разнесены по нескольким каталогам.

Следующая информация относится в первую очередь к связи по Ethernet. Если вы планируете использовать SLIP, прочитайте этот раздел, чтобы понять концепции, а затем обратитесь к специфическим для SLIP рекомендациям из последующего раздел.

Описание вашей сети Прежде, чем вы сможете описать (настроить) TCP/IP, вам необходимо определиться со следующуей информацией относительно установки сети.

- IP адрес. Это уникальный адрес машины в точечно-десятичном формате. Например, 128.253.153.54. Ваши сетевые администраторы снабдят вас таким номером.

  Если вы настраиваете только режим loopback (т.е. без SLIP, без карт Ethernet, только связь по TCP/IP внутри вашей машины) то ваш IP адрес будет 127.0.0.1.

- Маска вашей сети (“netmask”). Это ”точкосодержащий” квартет, похожий на IP адрес, определяющий, какая часть IP адреса относится к подсети, а какая относится к host (главной машине) этой подсети. (Если вас шокирует эта сетевая TCP/IP терминология - советуем почитать материалы по управлению сетями). Сетевая маска есть набор бит, который, будучи наложеным на адрес вашей сети, сообщит, к какой подсети относится этот адрес. Это очень важно для для маршрутизации (routing), и если вы обнаружите, например, что вы можете свободно общаться с людьми за пределами вашей сети, но не со своими соратниками внутри собственной сети, высока вероятность того, что вы неправильно задали маску.

  Администраторы вашей сети должны выбрать сетевую маску при проектировании сети, поэтому они могут сообщить вам правильную маску. Большинство сетей принадлежит классу C подсетей, которые используют сетевую маску 255.255.255.0. Другой класс сетей - B использует 255.255.0.0. Программы NET-2 автоматически выберут маску, которая предполагает отсутствие подсетей по умолчанию, поскольку иное вы не указали явно.

Это применимо также к порту loopback. Поскольку адрес порта loopback всегда 127.0.0.1, сетевая маска для этого порта всегда 255.0.0.0. Вы можете задавать это явно или полагаться на умолчание.

- Адрес вашей сети. Это ваш IP адрес с наложенной побитовой сетевой маской. Например, если ваша сетевая маска 255.255.255.0, а ваш IP адрес - 128.253.154.32, то адрес вашей сети - 128.253.154.0. А с сетевой маской 255.255.0.0 адрес вашей сети будет 128.253.0.0.

  Если вы используете только loopback, у вас нет адреса сети.

- Ваш бродкаст (broadcast - широковещательный) адрес. Бродкаст адрес используется для раздачи бродкаст пакетов на все машины вашей подсети. Поэтому, если хост-номера машинам вашей подсети даны по последним байтам IP-адресов (сетевая маска 255.255.255.0), ваш бродкаст адрес будет получен из вашего сетевого адреса наложением 0.0.0.255. Например, если ваш IP адрес 128.253.154.32 и ваша сетевая маска 255.255.255.0, то ваш бродкаст адрес 128.253.154.255.

  Чисто исторически сложилось, что некоторые сети настроены на использование сетевых адресов как бродкаст адресов. Если у вас возникнут сомнения, пообщайтесь с вашим сетевым администратором. (Во многих случаях бывает достаточно продублировать сетевую настройку других машин в вашей подсети, заменяя, разумеется IP адреса).

  Если только вы используете loopback, у вас не будет бродкаст адреса.

- Ваш шлюзовой (*gateway*) адрес. Это адрес машины, которая для вас является "шлюзом" во внешний мир (т.е. к машинам не вашей подсети). Во многих случаях шлюзовая машина имеет IP адрес, идентичный вашему, но с ".1" в качестве хост-адреса; т.е., если ваш IP адрес 128.253.154.32, ваш шлюз может быть 128.253.154.1. Ваш системный администратор даст вам IP адрес вашего шлюза.

  На самом деле, вы можете иметь несколько шлюзов. *Шлюз* - это просто машина, которая живет одновременно в двух различных сетях (имеет IP адреса различных подсетей) и маршрутизирует пакеты между ними. Многие сети имеют по одному шлюзу "во внешний мир" (к сети, непосредственно с вашей состыкованной), но в некоторых случаях у вас может быть несколько шлюзов в смежные сети.

  Если только вы пользуетесь loopback, у вас нет шлюзового адреса. То же самое имеет место, если у вас изолированная сеть.

- Адрес вашего сервера имен (nameserver). Большинство машин в сети имеют серверы имен, которые переводят имена хостов в IP адреса. Администратор вашей сети скажет адрес вашего сервера имен. Вы можете держать сервер на

> своей машине, используя `named`, в этом случае адрес сервера имен будет 127.0.0.1. Заводить сервер имен следует, только если у вас нет выбора, иначе выберите кого-то другого в сети, кто может это обеспечить. Настройка `named` это совсем другая песня; нам кажется, что вам на этом этапе лучше пообщаться с сетью. С именами вы можете разобраться позже.
>
> Если вы единственный имеете loopback, у вас нет адреса сервера имен.

Пользователи SLIP: Вышеприведенная информация может вам потребоваться, а может и не потребоваться. Разве что адрес сервера имен. При использовании SLIP, ваш IP адрес обычно определяется одним из двух способов: (a) У вас "статический" IP адрес, который не меняется в любое время выхода в сеть; (b) У вас "динамический" адрес, который берется из пула доступных адресов, когда вы связываетесь с сервером. В следующем разделе, посвященном настройке SLIP, это рассматривается более детально.

NET-2 поддерживает полную маршрутизацию, множественность маршрутов, обслуживание подсети (на этом этапе только в пределах байта). Выше описывались основные настройки TCP/IP. Ваши могут быть совсем другими: если есть сомнения, проконсультируйтесь у местных гуру из соседних сетей и посмотрите страницы Руководства про route и ifconfig. Настройка TCP/IP выходит далеко за рамки этой книги; вышенаписанного может быть достаточно большинству людей для начала.

rc-файлы в сети `rc`-файлы широко используемые в системе сценарии, выполняемые во время загрузки программой by `init`, которая запускает всех основных системных демонов (таких как `sendmail`, `cron`, и т.п.) и настраивает такие вещи, как сетевые параметры, системное хост-имя и т.п. rc-файлы обычно находятся в каталоге `/etc/rc.d`, но в других системах они могут быть в `/etc`.

Здесь мы собираемся описать rc-файлы, используемые при настройке TCP/IP. Файлов два: `rc.inet1` и `rc.inet2`. `rc.inet1` используется для настройки базовых сетевых параметров (таких как IP адреса и маршрутизация) и `rc.inet2` запускает TCP/IP демонов (`telnetd`, `tftpd` и т.д.).

Многие системы об'единяют оба этих файла в один, обычно называемый `rc.inet` или `rc.net`. Имена, данные вашим rc-файлам роли не играют, лишь бы они выполняли нужные функции и выполнялись во время загрузки программой `init`. Чтобы это обеспечить, возможно вам потребуется подредактировать `/etc/inittab` , чтобы выполнить соответствующие rc-файлы. В худшем случае вам придется создать `rc.inet1` и `rc.inet2` файлы заново и добавить информацию из `/etc/inittab`.

Как мы говорили, `rc.inet1` настраивает базовый сетевой интерфейс. Это включает ваше IP, сетевой адрес и таблицу маршрутизации (routing table) для вашей сети. Таблицы маршрутизации используются для маршрутизации входящих и исходящих сетевых дейтаграм (datagrams) на другие машины. Во многих простых настройках вы имеете три маршрута: один - для посылки пакетов своей собственной машине, другой - для посылки пакетов на другие машины вашей сети, третий - для посылки пакетов на машины, находящиеся за пределами вашей сети (через шлюзовую машину). Есть две программы для настройки этих параметров: `ifconfig` и `route`. Обе обычно находятся в `/etc` или `/sbin`.

`ifconfig` используется для настройки интерфейса устройств сети с необходимыми для функций параметрами, такими как IP адрес, маска сети, бродкаст адрес и т п. `route` используется для создания и модификации таблицы маршрутизации.

Для многих случаев файл `rc.inet1` подойдет в том виде, в каком он здесь приведен. Вы, разумеется, должны будете отредактировать его под свою систему. Не используйте без изменения IP и сетевой адреса, приведенные здесь в качестве примера, они соответствуют действительной машине в Internet.

```
#!/bin/sh
# This is /etc/rc.d/rc.inet1 -- Configure the TCP/IP interfaces

# First, configure the loopback device

HOSTNAME=`hostname`

/etc/ifconfig lo 127.0.0.1 # uses default netmask 255.0.0.0
/etc/route add 127.0.0.1 # a route to point to the loopback device

# Next, configure the ethernet device. If you´re only  using
# loopback or SLIP, comment out the rest of these lines.

# Edit for your setup.
IPADDR=”128.253.154.32”     # REPLACE with YOUR IP address
NETMASK=”255.255.255.0”     # REPLACE with YOUR netmask
NETWORK=”128.253.154.0”     # REPLACE  with  YOUR  network address
BROADCAST=”128.253.154.255” # REPLACE with YOUR  broadcast address,
                                  # if you have one. If not, leave blank
                                  # and edit below.
GATEWAY=”128.253.154.1”     # REPLACE with  YOUR  gateway address!

/etc/ifconfig eth0 $IPADDR netmask $NETMASK broadcast $BROADCAST

# If you don´t have a broadcast address, change the above line to:
# /etc/ifconfig eth0 $IPADDR netmask $NETMASK
```

```
/etc/route add $NETWORK

# The following is only necessary if you have a gateway; that is,
# your network is connected to the outside world.
/etc/route add default gw $GATEWAY metric 1

# End of Ethernet Configuration
```

Может вам придется немного поковырять этот файл, чтобы заставить работать. Этот сценарий должен подходить для настройки большинства простых сетей, но, разумеется, не для всех.

`rc.inet2` запускает различные сервера, используемые TCP/IP. Наиболее важный среди них - `inetd`. `inetd` сидит в фоне и присматривает за различными сетевыми портами. Когда машина пытается связаться с конкретным портом (например, со входным портом telnet), `inetd` создает новую копию соответствующего демона для этого порта (в случае порта telnet `inetd` запускает `in.telnetd`). Это проще, чем выполнять много независимых демонов (т.е. индивидуальных копий `telnetd`, `ftpd` и т.п.) - `inetd` запускает демонов только при возникновении необходимости.

`syslogd` - это системный демон входа - он аккумулирует сообщения о входе от различных источников и помещает их в log-файлы, зависящие от настройки `/etc/syslogd.conf`. `routed` - это сервер используемый для сопровождения динамической информации по маршрутизации. Когда ваша система пытается послать пакет в другую сеть, это может потребовать дополнительных записей в таблицу маршрутизации, чтобы это выполнить. `routed` заботится о сопровождении таблицы маршрутизации без необходимости вмешательства человека.

Приведенный ниже пример `rc.inet2` запускает лишь самый минимум серверов. Есть и другие серверы - многие из которых обслуживают настройки NFS. Пытаясь установить TCP/IP на вашей системе, обычно лучше всего начать с минимальной конфигурации и добавлять более сложные куски, (такие, как NFS), когда у вас уже что-то работает.

Обратите внимание, что в нижеприведенном файле мы предполагаем, что все сетевые демоны находятся в `/etc`. Ну и как обычно отредактируйте этот файл под свою конфигурацию.

```
#! /bin/sh
# Sample /etc/rc.d/rc.inet2

# Start syslogd
if [ -f /etc/syslogd ]
then
        /etc/syslogd
```

```
fi

# Start inetd
if [ -f /etc/inetd ]
then
        /etc/inetd
fi

# Start routed
if [ -f /etc/routed ]
then
        /etc/routed -q
fi

# Done!
```

Среди многих дополнительных серверов, которые вы можете запустить в `rc.inet2` - `named`. `named` - это сервер имен. Он отвечает за перевод (локальных) IP адресов в имена и наоборот. Если у вас где-нибудь в сети нет сервера имен или вы сами хотите снабжать локальными именами машин другие машины вашего домена, использование named необходимо. (Но для большинства случаев в этом нет необходимости). Настройка named достаточно сложна и требует предварительного планирования. Мы отсылаем заинтересованных читателей к хорошим книгам по TCP/IP.

/etc/hosts `/etc/hosts` содержит перечень IP адресов и имен хостов, которым они соответствуют. В общем, `/etc/hosts` содержат только записи для вашей локальной машины и, возможно, других "важных" машин (таких как сервер имен или шлюз). Перевод имя - адрес для других машин сети обеспечивает сервер имен.

Например,если ваша машина называется `loomer.vpizza.com` и имеет IP адрес 128.253.154.32, ваш `/etc/hosts` будет выглядеть как:

```
127.0.0.1           localhost
128.253.154.32      loomer.vpizza.com loomer
```

Если вы используете только loopback, единственная строка в `/etc/hosts` должна быть для 127.0.0.1 с именами `localhost` и хост-именем вашей машины.

/etc/networks Файл /etc/networks содержит ваши имена и адреса, а также других сетей. Он используется командой route и позволяет описывать сеть именами, если вы это захотите.

Всякая сеть, которую вы хотите добавить в маршрутизацию с использованием команды route (обычно вызываемой из `rc.inet1`) должна иметь запись в `/etc/networks`.

```
 Пример.

default 0.0.0.0 # default route    - mandatory
loopnet 127.0.0.0 # loopback network - mandatory
mynet 128.253.154.0 # Modify for your own network address
```

/etc/host.conf Чтобы обратиться к машине по домену, система должна определить ее физический адрес (IP-адрес). Система делает это либо находя соответствующий домен в файле `/etc/hosts` (см. руководство), либо обращаясь к специальным серверам, называемым серверами имен (nameserver). Файл `/etc/host.conf` задает Этот файл используется для описания порядка просмотра этих возможностей.

```
order hosts,bind
multi on
```

Эти строки указывают библиотекам разрешения вначале искать в файле `/etc/hosts` требуемый домен, а потом обратиться к серверу имен (если таковой имеется). Строка `multi` допускает множество IP адресов для одного имени машины в `/etc/hosts`.

/etc/resolv.conf Этот файл настраивает программы определения физического IP-адреса по домену машины, указывая IP-адрес вашего сервера имен и имя вашего домена. Имя вашего домена, это доменный адрес вашей машины в сети, с отброшенным именем машины. Так например, если ваше полное хост-имя `loomer.vpizza.com`, то имя вашего домена просто `vpizza.com`.

Например, если ваша машина `goober.norelco.com` и имеет сервер имен с адресом 128.253.154.5, ваш `/etc/resolv.conf` будет выглядеть:

```
domain     norelco.com
nameserver  127.253.154.5
```

Вы можете описать более одного сервера имен - каждый должен иметь свою строку в `resolv.conf`.

Установка хост-имени (hostname) Хост-имя устанавливается с помощью команды `hostname`. Она обычно вызывается из `/etc/rc` или `/etc/rc.local`; просто просмотрите свои системные rc-файлы, откуда вызывается. Например, если ваше (полное) хост-имя loomer.vpizza.com, отредактируйте соответствующий rc-файл, выполнив команду:

/bin/hostname loomer.vpizza.com

Обратите внимание, что команды `hostname` может не оказаться в /bin.

Попытка не пытка После того, как вы установили все эти файлы, вы должны быть готовы перезагрузить новое ядро и попытаться обрадоваться работающей сети. Правда, существует слишком много мест, где могут спрятаться ошибки, так что разумным будет проверить отдельные аспекты настройки сети (например, не самая хорошая идея для тестирования сети сразу шарахнуть по ней с помощью Mosaic с X-протоколом поверх IP).[48]

Вы можете использовать команду `netstat`, чтобы посмотреть таблицы маршрутизации; это обычно источник большинства неприятностей. Руководство по `netstat` описывает точный синтаксис этой команды в деталях. Для того, чтобы проверить связи в сети, мы предлагаем использовать клиента, такого, как `telnet`, чтобы связать машины вашей локальной подсети и внешней сети. Это позволит локализовать ошибки. (Например, если вы не можете связаться с локальной машиной, но связываетесь с машинами других сетей, скорее всего есть проблема с сетевой маской и настройкой таблицы маршрутизации). Вы можете также прямо вызвать команду `route` (под `root`) поэкспериментировать с записями таблицы маршрутизации.

Вам следует также проверить связи в сети прямо указывая IP адреса вместо хост-имен. Например, если у вас есть проблемы с командой

```
$ telnet shoop.vpizza.com
```

Причина может быть в некорректной настройке сервера имен. Попытайтесь использовать физический IP адрес машины; если это поможет, тогда вы будете знать, что ваши основные сетевые установки (скорее всего) правильны и проблема лежит в описании адреса сервера имен.

Отладка сетевых настроек может быть трудной задачей и мы не можем здесь втягиваться в ее обсуждение. Если вы не можете получить помощь от местных гуру, мы очень вам советуем почитать *Linux Network Administrators' Guide* из LDP.

---

[48] (прим. переводчика: сегодня бы автор вспомнил про Netscape)

5.3.3 Настройка SLIP

SLIP (Serial Line Internet Protocol) позволяет использовать TCP/IP на последовательных линиях, будь то коммутируемая телефонная линия с модемом или выделенная асинхронная линия. Разумеется, для использования SLIP вам необходим доступ к SLIP-серверу. Многие университеты и фирмы за умеренную плату предоставляют SLIP-вход.

Есть две основные программы, использующие SLIP: `dip` и `slattach`. Обе эти программы используются для установления SLIP-соединения через последовательные устройства. Необходимо использовать одну из этих программ, чтобы активизировать SLIP, недостаточно просто дозвониться до SLIP-сервера (с помощью коммуникационной программы вроде `kermit`) и запустить команды `ifconfig` и `route`, так как `dip` и `slattach` формируют специальный системный вызов `ioctl()`, чтобы перехватить управление последовательным устройством для реализации SLIP-интерфейса.

Dip может произвести дозвонку до SLIP-сервера, обеспечить соединение (handshaking) и войти на сервер (указав например, имя и пароль), а затем инициировать SLIP-соединение по последовательной линии. slattach же не делает ничего, кроме захвата устройства для использования его SLIP. Это полезно, если вы имеете постоянную линию для SLIP-сервера и нет необходимости в дозвонке и соединении для обеспечения связи. Но большинство пользователей предпочитает использовать `dip`.

Dip можно также использовать для настройки вашей системы Linux в качестве SLIP-сервера, когда другие машины к вам дозваниваются и выходят в сеть через вторичное соединение по Ethernet на вашей машине. Дополнительную информацию по этой процедуре смотрите в Руководстве на dip.

SLIP весьма отличается от Ethernet, в нем только две машины в ”сети” SLIP-хост (это вы) и SLIP-сервер. По этой причине SLIP часто воспринимается как связь ‘‘point-to-point’’ (от точки до точки). Обобщение этой идеи, известное как PPP (Point to Point Protocol) также реализовано в Linux.

Когда вы инициируете связь со SLIP-сервером, SLIP-сервер даст вам IP адрес. Некоторые SLIP-серверы выдают ”статические” IP адреса - в этом случае ваш IP адрес будет тот же самый всегда, когда вы связываетесь с сервером. Но большинство SLIP-серверов выдают IP адреса динамически, когда при каждой связи вы получаете IP адрес заново. В общем случае SLIP-сервер сообщит вам при установлении связи ваш IP адрес и адрес шлюза. dip способен читать эти значения при входе на SLIP-сервер и использовать их для настройки самого SLIP.

Существенное замечание. Настройка связи по SLIP похожа на настройку loopback или ethernet. Основные отличия обсуждаются ниже. Прочитайте предыдущий раздел про настройку базовых файлов TCP/IP, и выполните изменения, описанные ниже.

Соединение по dip при статическом IP адресе Если вы используете SLIP-сервер, выдающий статические IP адреса, вы можете включить записи о ваших IP адресе и хост-имени в `/etc/hosts`. А также настроить файлы, перечисленные в предыдущем разделе: `rc.inet2`, `host.conf` и `resolv.conf`. Также настроить `rc.inet1`, как описано выше. Если вы используете для связи со SLIP-сервером `dip`, то в файле `rc.inet1` для последовательного порта команды `ifconfig` и `route` вызывать не надо, `dip` вызовет эти команды после установления соединения. (Если же вы, используете `slattach`, вам будет необходимо включить команды `ifconfig` и route в `rc.inet1` для SLIP - смотрите ниже).

`dip` должен настраивать соответствующим образом таблицы маршрутизации для SLIP когда вы связываетесь. Однако, в некоторых случаях поведение `dip` может быть неправильным для ваших настроек и вам надо будет вручную выполнять команды `ifconfig` или `route` после того, как `dip` свяжется с сервером (это легче всего сделать из сценария shell, который содержит вызов `dip`, и немедленно выполнить соответствующие команды настройки). Ваш шлюз, это в большинстве случаев адрес SLIP-сервера. Вы можете знать этот адрес заранее или адрес шлюза будет выведен SLIP-сервером при установлении связи. Сценарий работы `dip` (описанный ниже) может также получать эту информацию от SLIP-сервера.

ifconfig может потребовать аргумента "pointopoint", если `dip` не настроил правильно интерфейс. Например, если адрес вашего SLIP-сервера 128.253.154.2, а ваш IP-адрес 128.253.154.32, вам может потребоваться выполнить команду под root

```
ifconfig sl0 128.253.154.32 pointopoint 128.253.154.2
```

после связи по dip.

Обратите внимание, что имена SLIP-устройств, используемые командами `ifconfig` и `route` - `sl0`, `sl1` и т.д.

В Разделе 5.3.4 мы объясним, как настраивать `dip` для связи со SLIP-сервером.

Соединение по slattach при статическом IP адресе Если у вас выделенная линия или кабель, идущий прямо к SLIP-серверу, то нет необходимости использовать `dip` для инициализации связи. Вместо этого может быть использована команда `slattach`. В этом случае ваш файл `/etc/rc.inet1` должен выглядеть примерно так:

```
#!/bin/sh
IPADDR="128.253.154.32"         # Replace with your IP address
REMADDR="128.253.154.2" # Replace with your SLIP server address

# Modify the following for the appropriate serial device for
# the SLIP connection:
slattach -p cslip -s 19200 /dev/ttyS0
/etc/ifconfig sl0 $IPADDR pointopoint $REMADDR up
/etc/route add default gw $REMADDR
```

`slattach` выделяет первое свободное SLIP-устройство (`sl0`, `sl1`, и т.д.) определенной последовательной линии.

Обратите внимание, что первый параметр команды `slattach` - это используемый SLIP-протокол. В настоящее время возможны только значения `slip` и `cslip`. `slip` - это обычный SLIP, как и следовало ожидать, а `cslip` - это SLIP с компрессией заголовков дейтаграмм. В большинстве случаев вам следует использовать `cslip`; однако, если у вас с ним возникают проблемы, попробуйте `slip`.

Если у вас более одного SLIP-интерфейса, то вы должны принять решения относительно маршрутизации. Вы должны решить, какие маршруты добавить, и эти решения могут быть сделаны только на базе действительного протокола связей вашей сети. Здесь вам могут помочь, как книга по TCP/IP, так и Руководство.

Соединение по dip при динамическом IP адресе Если ваш SLIP-сервер выдает IP адреса динамически, то вы, разумеется, не знаете заранее свой адрес, поэтому вы не можете включить его в `/etc/hosts`. (Между тем вы должны включить запись для своего хоста с адресом обратной связи (loopback address) 127.0.0.1.)

Многие SLIP-сервера выдают ваш IP адрес (также как и адрес сервера) во время соединения. Например, один тип SLIP-сервера выдает такое сообщение:

```
Your IP address is 128.253.154.44.
Server address is 128.253.154.2.
```

`dip` может перехватить эти номера с выхода сервера и использовать их для настройки SLIP-устройств.

Смотрите выше Раздел 5.3.3.1 относительно информации по настройке различных файлов для TCP/IP при использовании SLIP. Ниже мы об'ясняем, как настраивать `dip` для связи со SLIP-сервером.

5.3.4 Использование dip

`dip` может упростить процесс соединения со SLIP-сервером, войти и настроить SLIP-устройства. Если только у вас не выделенная линия для SLIP-сервера, `dip` - это то, что вам надо.

Для использования `dip` вы должны написать "сценарий болтовни" (‘‘chat script’’), который содержит перечень команд, используемых для связи со SLIP-сервером при входе в систему. Эти команды могут автоматически посылать ваши имя/пароль серверу, а также получать информацию о вашем IP адресе с сервера.

Вот пример такого сценария для использования с сервером динамических IP адресов. Для статических серверов вам потребуется в начале сценария установить значения переменных `$local` и `$remote`. В соответствии с вашими локальным IP адресом и адресом сервера соответственно. Более детальную информацию можно получить в Руководстве на `dip`.

```
main:
  # Set Maximum Transfer Unit. This is the maximum size of packets
  # transmitted on the SLIP device. Many SLIP servers use either
  # 1500 or 1006; check with your network admins when in doubt.
  get $mtu 1500

  # Make the SLIP route the default route on your system.
  default

  # Set the desired serial port and speed.
  port cua03
  speed 38400

  # Reset the modem and terminal line. If this causes trouble
  # for you, comment it out.
  reset

  # Prepare for dialing. Replace the following with your
  # modem initialization string.
  send AT&C1&D2N3&Q5%M3%C1N1W1L1S48=7r
  wait OK 2
  if $errlvl != 0 goto error
  # Dial the SLIP server
  dial 2546000
  if $errlvl != 0 goto error
  wait CONNECT 60
  if $errlvl != 0 goto error

  # We are connected.  Login to the system.
```

```
login:
  sleep 3
  send rnrn
  # Wait for the login prompt
  wait login: 10
  if $errlvl != 0 goto error

  # Send your username
  send USERNAMEn

  # Wait for password prompt
  wait ord: 5
  if $errlvl != 0 goto error

  # Send password.
  send PASSWORDn

  # Wait for SLIP server ready prompt
  wait annex: 30
  if $errlvl != 0 goto error

  # Send commands to SLIP server to initate connection.
  send slipn
  wait Annex 30

  # Get the remote IP address from the SLIP server. The
  # `get...remote´ command reads text in the form xxx.xxx.xxx.xxx,
  # and assigns it to the variable given as the second argument
  # (here, $remote).
  get $remote remote
  if $errlvl != 0 goto error
  wait Your 30

   # Get local IP address from SLIP server, assign to variable
   # $local.
  get $local remote
  if $errlvl != 0 goto error

  # Fire up the SLIP connection

done:
  print CONNECTED to $remote at $rmtip
  print GATEWAY address $rmtip
  print LOCAL address $local
  mode SLIP
  goto exit
error:
  print SLIP to $remote failed.
```

```
exit:
```

`dip` автоматически выполняет команды `ifconfig` и `route`, базирующиеся на значениях переменных `$local` и `$remote`. Здесь этим переменным присваиваются значения с использованием удаленных команд `get...`, которые получают текст со SLIP-сервера и присваивают его названной переменной.

Если команды `ifconfig` и `route`, которые выполняет для вас `dip` не работают, вы можете либо выполнить правильные команды в сценарии shell после выполнения `dip`, либо модифицировать исходник для самого `dip`. Выполнение `dip` с опцией `-v` будет выдавать отладочную информацию в процессе установления связи, что должно помочь в определении ошибок в работе. Теперь, для того, чтобы выполнить `dip` и открыть SLIP-соединение вы можете использовать команду, вроде:

```
/etc/dip/dip -v /etc/dip/mychat 2&1
```

Где различные dip-файлы и сценарий болтовни (`mychat.dip`) помещены в `/etc/dip`. Вышеприведенное обсуждение должно быть достаточным для вашего хорошего самочувствия на славном пути в сетевое сообщество через Ethernet или SLIP. И вновь мы настоятельно рекомендуем заглянуть в книгу по TCP/IP, особенно, если ваша сеть имеет специфику в маршрутизации, отличающую ее от рассмотренных здесь.

5.4 Сетевая работа с UUCP

UUCP (UNIX-to-UNIX Copy) - старейший механизм, используемый для передачи информации между системами UNIX. При использовании UUCP, системы UNIX созваниваются друг с другом (используя модем) и передают почтовые сообщения, новости, файлы и т.п. Если у вас нет TCP/IP или SLIP доступа, вы можете использовать для связи с миром UUCP. Большая часть программ, связанных с почтой и новостями (смотрите Разделы 5.5 и 5.6) может быть настроена на использование UUCP для передачи информации на другие машины. Действительно, если поблизости есть узел Internet, вы можете иметь доступ к почте Internet, получая ее с узла по UUCP.

Книга *Linux Network Administrator's Guide* содержит исчерпывающую информацию по настройке и использованию UUCP под Linux. Кроме того, по anonymous FTP `sunsite.unc.edu`, доступна *Linux UUCP HOWTO*, которая может быть весьма полезна. Другой источник информации по UUCP - книга Tim O'Reilly и Grace Todino *Managing UUCP and USENET*. Дополнительно смотрите Приложение A.

5.5 Электронная почта

Как и множество систем UNIX, Linux имеет несколько программных пакетов для использования электронной почты. E-mail (электронная почта) на вашей системе может быть либо локальная (то есть вы можете обмениваться почтой с другими пользователями вашей системы) или сетевая (то есть вы посылаете почту, используя либо TCP/IP, либо UUCP, другим пользователям). Программы e-mail обычно состоят из двух частей: *мэйлер* и *транспорт*. Мэйлер - это программы пользовательского уровня, которые используются для формирования и чтения почтовых сообщений. Популярные мэйлеры включают `elm` и `mailx`. Транспорт - это программы системного уровня, которые отвечают за доставку почты, как локальной, так и удаленной. Пользователь никогда не видит программы ”транспорт”; они взаимодействуют только с мэйлером. Но, назвавшись системным администратором, человек должен понимать концепции программ ”транспорта” и как их настраивать.

Наиболее популярная транспортная программа для Linux - это `Smail`. Эту программу просто настраивать. Она может посылать e-mail, локально и удаленно по TCP/IP и по UUCP. Yf В большинстве систем UNIX используется более мощная программа `sendmail`, однако, из-за сложного механизма установки, многие системы Linux ее не используют.

Linux Mail HOWTO дает больше информации относительно доступных почтовых программ для Linux и как их настраивать. Если вы планируете послать почту удаленному пользователю, вы должны понимать либо TCP/IP или UUCP, в зависимости от того, каким образом ваша машина подключена в сеть (смотри Разделы 5.3 и 5.4). Может быть полезной документация по UUCP и TCP/IP, перечисленная в Приложении A.

Большинство почтовых программ можно достать через anonymous FTP с `sunsite.unc.edu` в каталоге `/pub/Linux/system/Mail`.

5.6 Новости и USENET

Linux также обеспечивает ряд возможностей для работы с электронными новостями. При желании вы можете установить на вашей системе локальный сервер новостей, который позволит вам посылать ”статьи” (“articles”) в различные ”группы новостей” (“newsgroups”)... Удобная форма организации обсуждений. А если вы имеете выход по TCP/IP или UUCP в сеть, тогда вы будете в состоянии участвовать в USENET - всемирной сети новостей.

Программы новостей состоят из двух частей - *сервера* и *клиента*. *Сервер новостей* -

это программа, которая управляет группами новостей и занимается доставкой писем другим машинам (если вы в сети). Клиент новостей (*newsreader*) это программа, которая связывает с сервером, который позволяет пользователям получать и посылать новости.

Для Linux есть несколько серверов новостей. Они все имеют одни базовые протоколы и принципы. Две первые версии, это ‘‘C News’’ и ‘‘INN’’. Существует также много типов ”читалок” новостей (newsreaders), например `rn` и `tin`. Выбор читалки в той или иной мере дело вкуса. Все читалки работают одинаково с различными версиями серверных программ. Так что читалка независима от сервера и наоборот.

Если вы хотите лишь вести локальные новости (а не как часть USENET), то вам потребуется завести сервер на своей системе, а также инсталлировать читалку для пользователей. Сервер новостей будет хранить статьи в каталоге, например `/usr/spool/news`, а читалка будет их просматривать в поисках поступивших новостей.

Если вы захотите вести сетевые новости, вам предоставляется несколько возможностей. Новости, базирующиеся на сетевом TCP/IP, используют протокол, известный как NNTP (Network News Transmission Protocol). NNTP позволяет читалке читать новости прямо удаленно по сети. NNTP также позволяет серверам новостей посылать по сети статьи друг другу, это программа, на которой базируется USENET. Большинство фирм и университетов имеют один или более NNTP-серверов, установленых для работы со всеми новостями USENET данного узла. Каждая вторая машина на узле имеет базирующуюся на NNTP читалку для чтения и посылки новостей по сети через NNTP-сервер. Это означает, что только NNTP-сервер действительно хранит новости на диске.

Далее следует несколько сценариев настройки новостей.

- Вы ведете новости локально. То есть у вас нет в сеть выхода или желания возиться с сетевыми новостями. В этом случае вам надо выполнять C News или INN на вашей машине и инсталлировать читалку для чтения местных новостей.
- У вас есть выход по TCP/IP в сеть и на NNTP-сервер. Если ваша организация имеет NNTP-сервер новостей, вы можете читать и посылать новости с вашей Linux-машины, всего лишь инсталлировав у себя базирующуюся на NNTP читалку. (Большинство доступных читалок может быть настроено и на локальные новости и на использование NNTP). В этом случае вы не нуждаетесь в инсталляции сервера новостей или храните новости на своей системе. Читалка позаботится о чтении и посылке новостей по сети. Разумеется, вам потребуется настроенное TCP/IP и выход в сеть (смотрите Раздел 5.3).
- Вы имеете доступ к сети TCP/IP, но не имеете NNTP-сервера. В этом случае вы

можете использовать NNTP-сервер новостей на своей системе Linux. Вы можете инсталлировать либо локальную, либо базирующуюся на NNTP читалку и сервер будет помещать новости на вашу систему. В дополнение, вы можете настроить сервер для взаимодействия с другими NNTP-серверами новостей для передачи статей.

- Вы хотите передать новости, используя UUCP. Если у вас есть доступ по UUCP (смотри Раздел 5.4), вы также можете приобщиться к USENET. Вам будет необходимо инсталлировать (локальный) сервер новостей и программы чтения почты. Дополнительно, вам необходимо настроить вашу UUCP на периодическую передачу (прием) новостей на другую близлежащую UUCP машину (известную как "источник новостей" (“news feed”)). UUCP не использует передачу новостей по NNTP; просто у UUCP свой собственный механизм передачи новостей.

Темная сторона многих серверов новостей и читалок заключается в том, что они должны собираться вручную. Большинство программного обеспечения новостей не использует файлы настройки. Вместо этого опции настройки определяются при компиляции.

Большинство "стандартных" программ новостей (доступных через anonymous FTP с сервера `ftp.uu.net` каталог `/news`) - готовые для компиляции полуфабрикаты. Необходимые изменения (patches) можно найти на `sunsite.unc.edu` в `/pub/Linux/system/Mail` (который, совершенно случайно, находится там же, где Linux). Другие бинарные файлы программ новостей для Linux можно также найти в этом каталоге.

За дополнительной информацией обращайтесь к *Linux News HOWTO* на `sunsite.unc.edu` в `/pub/Linux/docs/HOWTO`. Кроме того, входящий в проект LDP *Linux Network Administrator's Guide* содержит исчерпывающую информацию по настройке программ новостей для Linux. Книга Tim O'Reilly и Grace Todino *Managing UUCP and Usenet* замечательное руководство по установке UUCP и программ новостей. Представляет интерес и документ USENET “How to become a USENET site”, доступный на `ftp.uu.net`, в каталоге `/usenet/news.announce.newusers`.

A Источники информации по Linux

Это приложение содержит информацию по различным источникам Linux, таким как документация, доступная по он-лайн, книги и другое. Многие из этих документов либо доступны в печатном виде, либо в электронном через Internet или BBS. Многие дистрибутивы Linux сами по себе также включают много документации, так что после инсталляции Linux эти файлы могут оказаться у вас в системе.

А.1 Документы, доступные по он-лайн

Эти документы должны быть доступны на любых FTP-серверах с архивами Linux (Смотри список в Приложении C). Если у вас нет прямого выхода по FTP, вы можете найти эти документы на других он-лайновых серверах (например, CompuServe, местные BBS и т.д.). Если у вас есть доступ к почте Internet, вы можете использовать сервис ftpmail для получения этих документов. Дополнительную информацию смотрите в Приложении C.

В частности, следующие документы могут быть найдены на `sunsite.unc.edu` в директории `/pub/Linux/docs`. Многие узлы имеют зеркальное отображение этого директория, однако, если у вас нет зеркального узла поблизости, можно связаться с названным ранее.

Вы можете также получить доступ к файлам Linux и документации с помощью `gopher`. Просто направьте вашего клиента gopher на порт 70 на `sunsite.unc.edu` и следуйте указаниям меню архива `Linux`. Это хороший способ интерактивного листания документации по Linux.

The Linux Frequently Asked Questions List (`http://sunsite.unc.edu/mdw/linux.html`)

The Linux Frequently Asked Questions List или “FAQ” (Список Часто Задаваемых Вопросов) - это список наиболее типовых вопросов (и ответов!) по Linux. Этот документ обеспечивает общей информацией по Linux, описывает типовые проблемы и решения, перечисляет другие источники информации. Каждому новому пользователю Linux следует прочитать этот документ. Он доступен в различных форматах, включая ”чистый” ASCII, PostScript, формат Lout. Linux FAQ сопровождается Ian Jackson, *ijackson@nyx.cs.du.edu* (`mailto:ijackson@nyx.cs.du.edu`).

The Linux META-FAQ (`http://sunsite.unc.edu/mdw/HOWTO/META-FAQ.html`)

The META-FAQ подборка ”матавопросов” по Linux; то есть про источники информации по Linux и другие вопросы общего характера. Это хорошее начало для абонента Internet, желающего получить больше информации о системе. Она сопровождается Michael K. Johnson, *johnsonm@sunsite.unc.edu* (`mailto:johnsonm@sunsite.unc.edu`).

The Linux INFO-SHEET (`http://sunsite.unc.edu/mdw/HOWTO/INFO-SHEET.html`)

The Linux INFO-SHEET техническое введение в систему Linux. Дается обзор системных характеристик и доступных программ. Также приводится перечень других источников информации про систему Linux. Формат и содержание похожи на META-FAQ; случайно похож и сопровождающий: Michael K. Johnson *johnsonm@sunsite.unc.edu* (`mailto:johnsonm@sunsite.unc.edu`).

The Linux Software Map (`http://www.boutell.com/lsm`)

The Linux Software Map - список многочисленных приложений, имеющихся для Linux; где их можно достать, кто их сопровождает и т.д. Он далек от полноты. - собрать полный список программ под Linux почти невозможно. Но этот список содержит большинство из наиболее популярных пакетов программ для Linux. Если вы не можете найти конкретные прикладные программы, удовлетворяющие ваши нужды, - хорошо начинать поиск с LSM. Он поддерживается Lars Wirzenius, *lars.wirzenius@helsinki.fi* (`mailto:lars.wirzenius@helsinki.fi`).

The Linux HOWTO Index (`http://sunsite.unc.edu/mdw/HOWTO/HOWTO-INDEX.html`)

The Linux HOWTO - собрание документов типа "как сделать", где каждый детально описывает конкретный аспект системы Linux. Он сопровождается Matt Welsh, *mdw@sunsite.unc.edu* (`mailto:mdw@sunsite.unc.edu`). (прим. переводчика: т.е. автором данной книги, если кто потерял мысль). The HOWTO Index перечисляет доступные документы HOWTO (некоторые из которых приведены ниже).

The Linux Installation HOWTO (`http://sunsite.unc.edu/mdw/HOWTO/Installation-HOWTO.html`

The Linux Installation HOWTO описывает, как получить и инсталлировать Linux, похоже на информацию, представленную в Главе 2.

The Linux Distribution HOWTO (`http://sunsite.unc.edu/mdw/HOWTO/Distribution-HOWTO.html`

Этот документ представляет перечень дистрибутивов Linux, которые можно получить по почте и через anonymous FTP. Он также включает информацию о других связанных с Linux вещах и сервисе. Приложение B содержит список поставщиков Linux, многие из которых перечислены в Distribution HOWTO.

The Linux XFree86 HOWTO (`http://sunsite.unc.edu/mdw/HOWTO/XFree86-HOWTO.html`)

Этот документ описывает, как инсталлировать и настраивать X Window System для Linux. Смотрите Раздел "5.1" где X Window System рассмотрена более детально.

The Linux Mail HOWTO (`http://sunsite.unc.edu/mdw/HOWTO/Mail-HOWTO.html`), *The Linux Ne*

Эти три документа описывают конфигурацию и установку электронной почты, новостей и UUCP на системе Linux. Поскольку эти три вопроса тесно переплетаются, вы можете читать их вместе.

The Linux Hardware HOWTO (`http://sunsite.unc.edu/mdw/HOWTO/Hardware-HOWTO.html`)

Этот документ содержит большой список аппаратуры, поддерживаемой Linux. Хотя этот список весьма далек от полноты, он может дать общее представление об аппаратуре, которая поддерживается системой.

The Linux SCSI HOWTO (`http://sunsite.unc.edu/mdw/HOWTO/SCSI-HOWTO.html`), *The Linux SC*

The Linux SCSI HOWTO полное руководство по настройке и использованию SCSI-устройств под Linux, таких как диски, ленты и CD-ROM.

The Linux NET-2-HOWTO (`http://sunsite.unc.edu/mdw/HOWTO/NET-2-HOWTO.html`)

The Linux NET-2-HOWTO описывает инсталляцию, установку и настройку программ "NET-2" TCP/IP под Linux, включая SLIP. Если вы хотите использовать TCP/IP на вашей системе Linux, вы должны прочитать этот документ.

The Linux Ethernet HOWTO (`http://sunsite.unc.edu/mdw/HOWTO/Ethernet-HOWTO.html`)

Документ тесно связан с NET-2-HOWTO. Ethernet HOWTO описывает различные устройства Ethernet, поддерживаемые Linux, и об'ясняет, как настраивать каждое из них при использовании Linux TCP/IP.

The Linux Printing HOWTO (`http://sunsite.unc.edu/mdw/HOWTO/Printing-HOWTO.html`), *The L*

Этот документ описывает, как настраивать печатающие устройства под Linux, такие как `lpr`. Настройка принтеров и программ печати под UNIX временами может быть очень непростой; этот документ проливает некоторый свет на эту проблему.

Приведенный ниже список отражает текущее состояние документов по HOWTO.

[49]

Linux Boot Prompt-Howto (http://sunsite.unc.edu/mdw/HOWTO/BootPrompt-HOWTO.html)
The Linux Bootdisk HOWTO (http://sunsite.unc.edu/mdw/HOWTO/Bootdisk-HOWTO.html)
The Linux Busmouse Howto (http://sunsite.unc.edu/mdw/HOWTO/Busmouse-HOWTO.html)
The Linux CD-ROM HOWTO (http://sunsite.unc.edu/mdw/HOWTO/CDROM-HOWTO.html)
Linux Commercial-HOWTO (http://sunsite.unc.edu/mdw/HOWTO/Commercial-HOWTO.html)
The Linux Cyrillic HOWTO (http://sunsite.unc.edu/mdw/HOWTO/Cyrillic-HOWTO.html)
The dosemu HOWTO (http://sunsite.unc.edu/mdw/HOWTO/DOSEMU-HOWTO.html)
The Linux Danish/International HOWTO (http://sunsite.unc.edu/mdw/HOWTO/Danish-HOWTO.html)
The Linux ELF HOWTO (http://sunsite.unc.edu/mdw/HOWTO/ELF-HOWTO.html)
Finnish-HOWTO (http://sunsite.unc.edu/mdw/HOWTO/Finnish-HOWTO.html)
Firewalling and Proxy Server HOWTO (http://sunsite.unc.edu/mdw/HOWTO/Firewall-HOWTO.html)
ftape-HOWTO (http://sunsite.unc.edu/mdw/HOWTO/Ftape-HOWTO.html)
The Linux GCC HOWTO (http://sunsite.unc.edu/mdw/HOWTO/GCC-HOWTO.html)
German-HOWTO: Anpassung von LINUX an deutsche Besonderheiten (http://sunsite.unc.edu/mdw/HOWTO/German-HOWTO.html)
Linux HAM-HOWTO, Amateur Radio. (http://sunsite.unc.edu/mdw/HOWTO/HAM-HOWTO.html)
The Hebrew HOWTO (http://sunsite.unc.edu/mdw/HOWTO/Hebrew-HOWTO.html)
Linux IPX-HOWTO (http://sunsite.unc.edu/mdw/HOWTO/IPX-HOWTO.html)
Linux Italian-HOWTO (http://sunsite.unc.edu/mdw/HOWTO/Italian-HOWTO.html)
Java on Linux HOWTO (http://sunsite.unc.edu/mdw/HOWTO/Java-HOWTO.html)
The Linux Kernel HOWTO (http://sunsite.unc.edu/mdw/HOWTO/Kernel-HOWTO.html)
The Linux Keyboard HOWTO (http://sunsite.unc.edu/mdw/HOWTO/Keyboard-HOWTO.html)
The MGR Window System HOWTO (http://sunsite.unc.edu/mdw/HOWTO/MGR-HOWTO.html)
The Linux NIS(YP)/NIS+/NYS HOWTO (http://sunsite.unc.edu/mdw/HOWTO/NIS-HOWTO.ht

---

[49] (прим. Переводчика: данный список отражает текущее состояние на момент перевода).

Linux PCI-HOWTO (`http://sunsite.unc.edu/mdw/HOWTO/PCI-HOWTO.html`)
Linux PCMCIA HOWTO (`http://sunsite.unc.edu/mdw/HOWTO/PCMCIA-HOWTO.html`)
Linux PPP HOWTO (`http://sunsite.unc.edu/mdw/HOWTO/PPP-HOWTO.html`)
Linux Portuguese-HOWTO (`http://sunsite.unc.edu/mdw/HOWTO/Portuguese-HOWTO.html`)
The Linux Serial HOWTO (`http://sunsite.unc.edu/mdw/HOWTO/Serial-HOWTO.html`)
Linux Shadow Password HOWTO (`http://sunsite.unc.edu/mdw/HOWTO/Shadow-Password-HOW`
The Linux Sound HOWTO (`http://sunsite.unc.edu/mdw/HOWTO/Sound-HOWTO.html`)
The Linux Sound Playing HOWTO (`http://sunsite.unc.edu/mdw/HOWTO/Sound-Playing-HOWT`
TERM HOWTO (`http://sunsite.unc.edu/mdw/HOWTO/Term-HOWTO.html`)
The Linux Tips HOWTO (`http://sunsite.unc.edu/mdw/HOWTO/Tips-HOWTO.html`)
UMSDOS HOW-TO (`http://sunsite.unc.edu/mdw/HOWTO/UMSDOS-HOWTO.html`)
The UPS Howto (`http://sunsite.unc.edu/mdw/HOWTO/UPS-HOWTO.html`)

Другие документы он-лайн

Если вы просмотрите поддиректории с документами на любом Linux FTP-сервере, вы увидите много других документов, которые здесь не перечислены: множество FAQ, интересных новостей и другой важной информации. Эту россыпь трудно здесь классифицировать; если вы не найдете, что искали, посмотрите еще в архивах Linux серверов, перечисленных в Приложении С.

А.2 Руководства проекта LDP (Linux Documentation Project)

В рамках проекта The Linux Documentation Project (LDP) существует и разрабатывается множество руководств и другой документации для Linux, включая Руководство (информация, содержащаяся в команде "man"). Эти руководства находятся в различной стадии готовности и мы с благодарностью принимаем любую помощь по их правке, обновлению, совершенствованию. Если у вас есть вопросы по поводу LDP, пожалуйста свяжитесь с Matt Welsh *mdw@sunsite.unc.edu* (`mailto:mdw@sunsite.unc.edu`).

Эту книгу наанглийскомязыке можно получить через anonymous FTP из многих архивов Linux, включая `sunsite.unc.edu` в директории `/pub/Linux/docs/LDP`. Ряд дистрибуторов продает печатные копии этой книги. В будущем вы сможете найти руководства LDP на полках ваших книжных магазинов.[50]

Linux Installation and Getting Started, by Matt Welsh (`http://sunsite.unc.edu/mdw/LDP/gs/gs.h`

---

[50] (прим. переводчика: за рубежом, в той же Америке, это будущее уже давно наступило).

Новое руководство пользователя по Linux, содержащее всю необходимую информацию для начала работы. Может так случиться, что вы держите эту книгу в руках.

The Linux System Administrators' Guide, by Lars Wirzenius (`http://linuxwww.db.erau.edu/SAG`)

Это полное руководство по эксплуатации и настройке системы Linux. Существует много специфических для Linux особенностей в работе Администратора системы, таких как поддержка сообщества пользователей, сопровождение файловой системы, резервирование системы и другое. Все это обсуждается в данном руководстве.

The Linux Network Administrators' Guide, by Olaf Kirch (`http://linuxwww.db.erau.edu/NAG`)

Подробное и полное руководство по сетевой работе под Linux, включая TCP/IP, UUCP, SLIP и т.п. Эта книга - очень хорошее чтение. Она содержит много ценной информации по многим аспектам, проясняет многие сложные вопросы настройки и эксплуатауции сети.

The Linux Kernel Hackers' Guide, by Michael Johnson (`http://linuxwww.db.erau.edu/KHG`)

Интересные детали относительно ядра и разработок под Linux. Linux уникален тем, что доступны все исходные тексты ядра. Эта книга открывает двери разработчикам, которые желали бы заняться расширением и модификацией ядра. Это руководство также содержит понятное описание концепций ядра и используемых в Linux соглашений.

Приведенные ниже документы входят в проект LDP на момент перевода

Programmer's Guide (`http://linuxwww.db.erau.edu/LPG`)
User's Guide (`http://linuxwww.db.erau.edu/LUG`)

A.3 Книги и другие публикации

Linux Journal (`http://www.ssc.com/lj/index.html`) - ежемесячный журнал для и про сплотившееся вокруг Linux сообщество, который наполняют и выпускают разработчики и энтузиасты Linux. Он распространяется по всему миру и является хорошим средством поддержания непосредственного контакта с динамическим миром Linux, особенно если у вас нет выхода в USENET.

В момент написания этой книги подписка на Linux Journal стоила $19 в год в США, $24 в Канаде и US$29 в остальном мире. По поводу подписки или для получения дополнительной информации пишите в Linux Journal, PO Box 85867, Seattle, WA, 98145-1867, USA, или звоните +1 206 527-3385. Номер их факса +1 206 527-2806, и e-mail linux@ssc.com. Вы можете также найти Linux Journal FAQ и некоторые статьи через anonymous FTP на `sunsite.unc.edu` в `/pub/Linux/docs/linux-journal`.

Как мы уже говорили, мало книг было опубликовано непосредственно по Linux. Так что, если вы новичок в мире UNIX или хотите получить дополнительную информацию, кроме представленной здесь, мы рекомендуем посмотреть следующие доступные книги.

A.3.1 Использование UNIX

Название:	Learning the UNIX Operating System
Автор:	Grace Todino & John Strang
Издатель:	O´Reilly and Associates, 1987
ISBN:	0-937175-16-1, $9.00

Хорошая книга для начинающего изучать операционную систему UNIX. Большая часть информации может быть применена также к Linux. Я рекомендую эту книгу тем, кто является новичком и действительно желает начать пользоваться этой новой для него системой.

Название:	Learning the vi Editor
Автор:	Linda Lamb
Издатель:	O´Reilly and Associates, 1990
ISBN:	0-937175-67-6, $21.95

Эта книга о редакторе vi - мощном текстовом редакторе, который можно найти на любой системе UNIX мира. Часто бывает важно знать и быть готовым применить vi, поскольку не всегда вы будете иметь доступ к "настоящим" редакторам, вроде Emacs.

A.3.2 Системное администрирование

Название:	Essential System Administration
Автор:	AEleen Frisch
Издатель:	O´Reilly and Associates, 1991
ISBN:	0-937175-80-3, $29.95

Из каталога O'Reilly and Associates: "Как любая другая многопользовательская система, UNIX требует определенной заботы и внимания. Essential System Administration и рассказывает как это надо делать. Эта книга развеивает миф и замешательство вокруг этого вопрос; дает краткое информативное введение в задачи, с которыми сталкивается любой, отвечающий за эксплуатацию системы UNIX". Я лучше этого сформулировать не могу.

Название: TCP/IP Network Administration
Автор: Craig Hunt
Издатель: O'Reilly and Associates, 1990
ISBN: 0-937175-82-X, $24.95

Полное руководство по установке и эксплуатации TCP/IP. Хотя эта книга не посвящена непосредственно Linux, приблизительно 90% ее применимо к Linux. Совместно с Linux NET-2-HOWTO и Linux Network Administrator's Guide это замечательная книга, обсуждающая соответствующие концепции и детали работы с TCP/IP.

Название: Managing UUCP and Usenet
Автор: Tim O'Reilly and Grace Todino
Издатель: O'Reilly and Associates, 1991
ISBN: 0-937175-93-5, $24.95

Эта книга рассматривает вопросы, как инсталлировать и настраивать сетевые программы UUCP, включая настройку на работу с новостями USENET. Если вас интересует использование UUCP или доступ к новостям USENET, вам следует прочитать эту книгу.

A.3.3 The X Window System

Название: The X Window System: A User's Guide
Автор: Niall Mansfield
Издатель: Addison-Wesley
ISBN: 0-201-51341-2, ??

Полное учебное пособие и справочник по использованию X Window System. Если вы инсталлируете X windows на вашем Linux и хотите знать, как извлечь из этого максимум, вам стоит прочитать эту книгу. В отличие от других оконных систем, многие возможности, которые дает X не видны с первого взгляда.

A.3.4 Программирование

Название: The C Programming Language
Автор: Brian Kernighan and Dennis Ritchie
Издатель: Prentice-Hall, 1988
ISBN: 0-13-110362-8, $25.00

Эту книгу следует иметь обязательно любому, пожелавшему программировать на C в ОС UNIX. (Или в другой системе из этой обоймы). Хотя эта книга не ограничивается лишь UNIX, она вполне применима к программированию на C в UNIX.

Название: The Unix Programming Environment
Автор: Brian Kernighan and Bob Pike
Издатель: Prentice-Hall, 1984
ISBN: 0-13-937681-X, ??

Обзор программирования в UNIX. Рассматриваются все инструменты; хорошее чтение для желающих познакомиться с аморфным миром UNIX.

Название: Advanced Programming in the UNIX Environment
Автор: W. Richard Stevens
Издатель: Addison-Wesley
ISBN: 0-201-56317-7, $50.00

Это мощный том, содержащий все, что вам необходимо знать для программирования на системном уровне UNIX - ввод-вывод файлов, процессы, управление, взаимодействие процессов, сигналы, работа с терминалом. Эта книга делает акцент на различных стандартах UNIX, включая POSIX.1, который в значительной степени выдерживается в Linux.

A.3.5 Kernel Hacking

Название: The Design of the UNIX Operating System
Автор: Maurice J. Bach
Издатель: Prentice-Hall, 1986
ISBN: 0-13-201799-7, ??

Эта книга рассматривает алгоритмы и внутренние механизмы ядра UNIX. Она не привязана к какому-то конкретно ядру, хотя больше тяготеет к System V-isms. Это лучшее, с чего можно начать, если хотите понять, что и как ”тикает” внутри системы Linux.

Название: The Magic Garden Explained
Автор: Berny Goodheart and James Cox
Издатель: Prentice-Hall, 1994
ISBN: 0-13-098138-9, ??

Эта книга детально описывает ядро System V R4. В отличие от книги Bach-а, которая прежде всего сосредотачивается на алгоритмах, которые ”тикают” в ядре, эта книга представляет реализацию SVR4 на более техническом уровне. Хотя Linux и SVR4 дальние родственники, эта книга может дать представление о действительной работе реального ядра UNIX. Кроме того, это достаточно свежая книга, посвященная ядру UNIX - издана в 1994.

B Список поставщиков Linux

Это приложение содержит информацию для контактов с продавцами Linux на дискетах, ленте и CD-ROM'ах. Многие из них поддерживают документацию по Linux и другой сервис. Без сомнения, этот список не полон. Если вы читаете эту книгу с твердой копии, не исключено что продавец или издательство поддерживают Linux.

Автор не гарантирует полную точность информации, приведенной ниже.
Fintronic Linux Systems
1360 Willow Rd., Suite 205
Menlo Park, CA 94025 USA
Tel: +1 415 325-4474
Fax: +1 415 325-4908
linux@fintronic.com (`mailto:linux@fintronic.com`)

InfoMagic, Inc.
PO Box 30370
Flagstaff, AZ 86003-0370 USA
Tel: +1 800 800-6613, +1 602 526-9565
Fax: +1 602 526-9573
Orders@InfoMagic.com (`mailto:Orders@InfoMagic.com`)

Lasermoon Ltd
2a Beaconsfield Road, Fareham,
Hants, England. PO16 0QB.
Tel: +44 (0) 329 826444.
Fax: +44 (0) 329 825936.
info@lasermoon.co.uk (`mailto:info@lasermoon.co.uk`)

Linux Journal
P.O. Box 85867
Seattle, WA 98145-1867 USA
Tel: +1 206 527-3385
Fax: +1 206 527-2806
linux@ssc.com (`mailto:linux@ssc.com`)

Linux Systems Labs
18300 Tara Drive
Clinton Twp, MI 48036 USA
Tel: +1 313 954-2829, +1 800 432-0556
Fax: +1 313 954-2806
info@lsl.com (`mailto:info@lsl.com`)

Morse Telecommunication, Inc.
26 East Park Avenue, Suite 240
Long Beach, NY 11561 USA
Tel: +1 800 60-MORSE
Fax: +1 516 889-8665
Linux@morse.net (`mailto:Linux@morse.net`)

Nascent Technology
Linux from Nascent CDROM
P.O. Box 60669
Sunnyvale CA 94088-0669 USA
Tel: +1 408 737-9500
Fax: +1 408 241-9390
nascent@netcom.com (`mailto:nascent@netcom.com`)

Red Hat Software
P.O. Box 4325
Chapel Hill, NC 27515 USA
Tel: +1 919 309-9560
redhat@redhat.com (`mailto:redhat@redhat.com`)

SW Technology
251 West Renner Suite 229
Richardson, TX 75080 USA
Tel: +1 214 907-0871

swt@netcom.com (`mailto:swt@netcom.com`)

Takelap Systems Ltd.
The Reddings, Court Robin Lane,
Llangwm, Usk, Gwent, United Kingdom NP5 1ET.
Tel: +44 (0)291 650357
Fax: +44 (0)291 650500
info@ddrive.demon.co.uk (`mailto:info@ddrive.demon.co.uk`)

Trans-Ameritech Enterprises, Inc.
2342A Walsh Ave
Santa Clara, CA 95051 USA
Tel: +1 408 727-3883
roman@trans-ameritech.com (`mailto:roman@trans-ameritech.com`)

Unifix Software GmbH
Postfach 4918
D-38039 Braunschweig
Germany
Tel: +49 (0)531 515161
Fax: +49 (0)531 515162

Yggdrasil Computing, Incorporated
4880 Stevens Creek Blvd., Suite 205
San Jose, CA 95129-1034 USA
Tel: +1 800 261-6630, +1 408 261-6630
Fax: +1 408 261-6631
info@yggdrasil.com (`mailto:info@yggdrasil.com`)

C Учебные материалы по FTP и список серверов

FTP (“File Transfer Protocol”) - Протокол Передачи Файлов, это множество программ, используемых для передачи файлов между системами в Internet. Большинство систем UNIX, VMS и MS-DOS в Internet имеют программу, называемую ftp, которая используется для передачи этих файлов и, если у вас есть выход в Internet, лучший способ скачать программы для Linux, это с помощью `ftp`. Это приложение рассматривает основы использования `ftp`. Разумеется, `ftp` имеет значительно больше функциональных возможностей, чем рассмотрено здесь.

В конце этого приложения дан список FTP-серверов, где можно найти программы для Linux. Кроме того, если у вас нет прямого выхода в Internet, но вы можете обмениваться с Internet электронной почтой, то ниже приводится и информация по использованию сервиса `ftpmail`.

Если вы используете системы MS-DOS, UNIX или VMS для скачивания файлов из Internet, то для вас ftp - это программа, управляемая командами. В то время как в других реализациях ftp, таких как версия для Macintosh (называемая Fetch), имеются для этого чудесные меню, которые самопонятны. Даже если вы не используете командно-управляемое `ftp`, информация, приводимая здесь, все равно может быть полезной.

ftp можно использовать как для посылки файлов (upload), так и для их получения (download) с других узлов (sites) Internet.[51]

В большинстве случаев вы все-таки будете скачивать программы. В Internet большое количество общедоступных архивных FTP-серверов, т.е. машин, позволяющих любому войти на них по `ftp` и скачать свободнораспространяемое (free)[52] программное обеспечение. Один из таких архивных серверов `sunsite.unc.edu`, который состоит из множества Санов (Sun Microsystems), работает как (единый) один из самых мощных серверов Linux. Кроме того, архивные FTP-сервера осуществляют ”зеркальное” отображение друг на друга, то есть, размещенное на одном сервере, автоматически копируется на ряд других серверов. так что не удивляйтесь, если увидите один и тот же файл на многих различных архивных серверах.

C.1 Начала ftp

Обратите внимание на то, что в ”экранах”, приводимых ниже, я лишь показываю наиболее существенную информацию, так что то, что вы можете увидеть на реальном экране, может отличаться.

[51] (прим. переводчика: нет общепринятого перевода слова ”site”, которое, следуя скорее традиции Relcom, мы в зависимости от контекста переводим, то как ”узел”, то как ”(сетевой, архивный) сервер”; по этой же причине, часто для ”download” используется (в том числе и нами) термин ”скачать”, что, видимо, следует признать (техническим) жаргоном).

[52] (прим. переводчика: сравните два последних слова и простите переводчику использование небольшой дозы жаргона)

Для запуска `ftp` и установления связи с сервером просто используйте команду

```
ftp hostname
```

где <`hostname`> - имя сервера, с которым вы связываетесь. Например, для связи с мифическим сервером `shoop.vpizza.com` можно использовать команду

```
ftp shoop.vpizza.com
```

C.2 Вход на сервер

Когда запускается `ftp`, мы можем увидеть что-то вроде

```
Connected to shoop.vpizza.com.
220 Shoop.vpizza.com FTPD ready at 15 Dec 1992 08:20:42 EDT
Name (shoop.vpizza.com:mdw):
```

Здесь `ftp` просит нас ввести имя (Name) пользователя, под которым мы хотим войти на сервер `shoop.vpizza.com`. По умолчанию здесь "`mdw`", что служит моим именем пользователя для FTP-входа. Поскольку у меня нет account[53] на `shoop.vpizza.com`, я не могу войти под своим именем. Вместо этого, чтобы войти на общедоступный FTP-сервер вы входите как `anonymous` (аноним) и сообщаете свой адрес e-mail (если он у вас есть) в качестве пароля. То есть нам следует ввести

```
Name (shoop.vpizza.com:mdw):  anonymous
331-Guest login ok, send e-mail address as password.
Password:  mdw@sunsite.unc.edu
230- Welcome to shoop.vpizza.com.
230- Virtual Pizza Delivery[tm]:  Download pizza in  30 cycles
or less
230- or you get it FREE!
ftp
```

Разумеется, вам следует сообщать свой e-mail адрес вместо моего, он не будет отображаться на экране при вашем вводе (поскольку он вводится под "вывеской" пароля). `ftp` должен позволить нам войти и мы будем иметь возможность скачивать программы.

[53] (прим. переводчика: не открыт счет, а точнее (но дальше от текста) - не зарегистрирован в качестве пользователя)

C.3 Озираясь вокруг

О'кей, мы вошли. `ftp>` - это наша подсказка, теперь программа `ftp` ждет ввода команд. Есть несколько основных команд, которые вам надо знать. Прежде всего, команды

```
ls file
```

и

```
dir file
```

обе дают список файлов (где `<file>` - необязательный аргумент, указывающий, какой список вывести). Разница в том, что `ls` обычно выдает короткий список, а `dir` - длинный (то есть с большей информацией относительно размера файлов, даты модификации и т.п.). Команда

```
cd directory
```

переместит "вас" в указанный каталог (точно также, как команда `cd` в UNIX или MS-DOS). Вы можете использовать команду

```
cdup
```

для перехода в родительский (находящийся выше) каталог

Команда

```
help command
```

даст вам подсказку по указанной команде `ftp` (такой как `ls` или `cd`). Если команда не указана, `ftp` выдаст список всех доступных команд. Если мы введем теперь `dir`, мы увидим начальный каталог нашего местонахождения.

```
ftp dir
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
total 1337

dr-xr-xr-x  2 root     wheel         512 Aug 13 13:55 bin
drwxr-xr-x  2 root     wheel         512 Aug 13 13:58 dev
drwxr-xr-x  2 root     wheel         512 Jan 25 17:35 etc
drwxr-xr-x 19 root     wheel        1024 Jan 27 21:39 pub
drwxrwx-wx  4 root     ftp-admi     1024 Feb  6 22:10 uploads
```

```
drwxr-xr-x  3 root     wheel         512 Mar 11  1992 usr

226 Transfer complete.
921 bytes received in 0.24 seconds (3.7 Kbytes/s)
ftp
```

Каждая из этих строк представляет каталог, а не отдельный файл, который мы можем скачать (на это указывает символ "`d`" в первой позиции списка). На большинстве архивных FTP-серверов общедоступные программы находятся под каталогом `/pub`, так что давайте туда и пойдем.

```
ftp cd pub
ftp dir
200 PORT command successful.
150 ASCII data connection  for /bin/ls (128.84.181.1,4525)  (0
bytes).
total 846

-rw-r--r--  1 root   staff     1433 Jul 12  1988 README
-r--r--r--  1 3807   staff    15586 May 13  1991  US-DOMAIN.TXT.2
-rw-r--r--  1 539    staff    52664 Feb 20  1991 altenergy.avail
-r--r--r--  1 65534  65534    56456 Dec 17  1990 ataxx.tar.Z
-rw-r--r--  1 root   other  2013041 Jul  3  1991 gesyps.tar.Z
-rw-r--r--  1 432    staff    41831 Jan 30  1989 gnexe.arc
-rw-rw-rw-  1 615    staff    50315 Apr 16  1992 linpack.tar.Z
-r--r--r--  1 root   wheel    12168 Dec 25  1990 localtime.o
-rw-r--r--  1 root   staff     7035 Aug 27  1986 manualslist.tblms
drwxr-xr-x  2 2195   staff      512 Mar 10 00:48 mdw
-rw-r--r--  1 root   staff     5593 Jul 19  1988 t.out.h

226 ASCII Transfer complete.
2443 bytes received in 0.35 seconds (6.8 Kbytes/s)
ftp
```

Здесь мы можем видеть ряд (интересных?) файлов, один из которых называется `README`, который нам следует скачать (большинство FTP-серверов имеет файл `README` в каталоге /pub).

C.4 Скачивание файлов

Перед скачиванием файлов следует обратить внимание на

- Включите вывод хэш-меток. Хэш-метки выводятся на экран как передаваемые файлы; они информируют о переданных об'емах и что передача не зависла (так что вы не будете сидеть минут 20, полагая, что вы все еще скачиваете файл). В общем случае хэш-метка появляется, как символ решетки (#), который печатается через каждые 1024 или 8192 переданных байт, в зависимости от системы.

Включение хэш-меток осуществляется командой

```
ftp hash
Hash mark printing on (8192 bytes/hash mark).
ftp
```

- Определите тип файла, который вы скачиваете. FTP различает два типа файлов: бинарные (двоичные) и текстовые. Большинство файлов, которые вы будете скачивать - бинарные. Это программы, скомпрессированные файлы, архивированные файлы и т.п. Но есть немало и текстовых файлов (вроде `README`).

  Почему тип файла имеет значение? Только потому, что на некоторых системах (таких, как MS-DOS), определенные символы текстового файла, вроде возврата каретки, должны быть преобразованы так, чтобы файл был читаем. А при передаче бинарных файлов никаких преобразований не осуществляется - файл просто передается байт за байтом.

  Команды `bin` и `ascii` используются для перевода соответственно в бинарный режим передачи файлов и текстовый соответственно. Если возникают сомнения - используйте бинарный режим. Если вы попытаетесь передать бинарный файл в текстовом режиме, файл будет испорчен при передаче до полной бесполезности. (Это одна из популярнейших ошибок на начальных этапах использования FTP). Но разумеется, вы можете использовать текстовый режим для передачи нормальных текстовых файлов (чьи имена часто заканчиваются на `.txt`).

  Например, мы скачиваем файл README который, скорее всего, текстовый так что мы используем команду

```
ftp ascii
200 Type set to A.
ftp
```

- Установите свой местный каталог. Ваш местный каталог, это каталог вашей системы, куда вы хотите в конечном счете скачать файлы. В то время как команда cd меняет каталог удаленной машины (машины, на которую вы вошли по FTP), команда `lcd` меняет местный (l-local) каталог.

  Например, чтобы установить местный каталог `/home/db/mdw/tmp`, используйте команду

```
ftp lcd /home/db/mdw/tmp
```

```
          Local directory now /home/db/mdw/tmp
          ftp
```

Теперь вы действительно готовы скачивать файлы. Команда

```
get remote-name local-name
```

используется именно для этого, где <`remote-name`> имя файла на удаленной машине, а <`local-name`> - имя, которое вы хотите дать файлу на вашей машине. <`local-name`> - необязательный аргумент; по умолчанию имя местного файла то же, что и у скачиваемого файла. Но, если например вы скачиваете файл `README`, и у вас уже есть файл с именем `README` в этом каталоге, вам следует дать другое имя для <`local-filename`>, чтобы не затереть существующий. Например, для скачивания файла `README` мы просто используем

```
ftp get README
200 PORT command successful.
150 ASCII data connection for README (128.84.181.1,4527) (1433
bytes).
#
226 ASCII Transfer complete.
local:  README remote:  README
1493 bytes received in 0.03 seconds (49 Kbytes/s)
ftp
```

C.5 Прекращение FTP-связи

Для прекращения FTP-сессии просто используйте команду

```
quit
```

Команда

```
close
```

может использоваться для закрытия связи с данным FTP-сервером; команда `open` может быть после этого использована для начала сессии с другим сервером (без выхода из программы FTP).

```
ftp close
221 Goodbye.
ftp quit
```

C.6 Использование ftpmail

`ftpmail` - это сервис, который позволяет получить файлы из FTP-архива через электронную почту Internet. Если у вас нет прямого выхода на Internet, но есть возможность посылать письма в Internet (например, из CompuServe), ftpmail - это хороший способ получить файлы из архивов FTP-серверов. К сожалению, `ftpmail` может быть медленным, особенно при пересылке больших об'емов. Перед попыткой скачать большое число программ с использованием `ftpmail`, убедитесь, что область спуллинга вашей mail будет способна справиться с такими об'емами входного трафика. Во многих системах есть ограничения на об'ем приходящей электронной почты[54] Так что руководствуйтесь здравым смыслом.

`sunsite.unc.edu` - один из главных архивных Linux FTP-серверов, имеет также `ftpmail`-сервер. Для того, чтобы воспользоваться этим сервисом, пошлите письмо по адресу:

```
ftpmail@sunsite.unc.edu
```

А в письме только одно слово

```
help
```

В результате вы получите в ответ список команд `ftpmail` и краткую инструкцию по пользованию системой.

Например, для получения списка файлов sunsite.unc.edu, пошлите письмо по вышеуказанному адресу со следующим текстом:

```
open sunsite.unc.edu
cd /pub/Linux
dir
quit
```

Вы можете использовать `ftpmail` для связи с архивом FTP-сервера; и это не ограничивается лишь `sunsite.unc.edu`. В следующем разделе приводится список Linux FTP-архивов.

C.7 Список Linux FTP-серверов

Таблица C.1 - это список наиболее известных архивных FTP-серверов, которые хранят пакеты программ для Linux. Имейте в виду, что многие другие сервера имеют

---

[54] (прим. переводчика: ограничения могут быть как на об'ем отдельного файла, так и на общее число принимаемых (размещаемых во входном спуле) файлов).

зеркальные отображения многих из этих архивов, так что скорее всего вы найдете нужное вам на серверах, отсутствующих в этом списке.

`tsx-11.mit.edu`, `sunsite.unc.edu` и `nic.funet.fi` - это "дом родной" для Linux, где вы можете найти большинство новых программ. Большинство других серверов из списка зеркально отображают некоторую комбинацию из этих трех. Для уменьшения сетевого трафика выбирайте сервер, который ближе к вам географически.

Имя сервера	IP адрес	Каталог
tsx-11.mit.edu	18.172.1.2	/pub/linux
sunsite.unc.edu	152.2.22.81	/pub/Linux
nic.funet.fi	128.214.6.100	/pub/OS/Linux
ftp.mcc.ac.uk	130.88.200.7	/pub/linux
fgb1.fgb.mw.tu-muenchen.de	129.187.200.1	/pub/linux
ftp.informatik.tu-muenchen.de	131.159.0.110	/pub/Linux
ftp.dfv.rwth-aachen.de	137.226.4.105	/pub/linux
ftp.informatik.rwth-aachen.de	137.226.112.172	/pub/Linux
ftp.ibp.fr	132.227.60.2	/pub/linux
kirk.bu.oz.au	131.244.1.1	/pub/OS/Linux
ftp.uu.net	137.39.1.9	/systems/unix/linux
wuarchive.wustl.edu	128.252.135.4	/systems/linux
ftp.win.tue.nl	131.155.70.100	/pub/linux
ftp.ibr.cs.tu-bs.de	134.169.34.15	/pub/os/linux
ftp.denet.dk	129.142.6.74	/pub/OS/linux

Таблица C.1: FTP-серверы Linux

D Список BBS для Linux

Приведенний ниже список перечисляет электронные доски об'явлений (bulletin board systems - BBS), содержащих программное обеспечение по Linux.

Этот список поддерживает *Zane Healy* (`mailto:healyzh@holonet.net`). Если вы знаете или ведете BBS, не включенные в данный список свяжитесь с ним.

В Linux-сообщество входят не только пользователи Internet. В действительности, большинство Linux пользователей не имет (пока) доступа к Internet. Так что очень важно, что BBS продолжают поддерживать Linux по всему миру.

D.1 США

`Citrus Grove Public Access, 916-381-5822. ZyXEL 16.8/14.4 Sacramento,` CA. Internet: citrus.sac.ca.us

Higher Powered BBS, 408-737-7040. ? CA. RIME -HIGHER

hip-hop, 408-773-0768. 19.2k Sunnyvale, CA. USENET access

hip-hop, 408-773-0768. 38.4k Sunnyvale, CA.

Unix Online, 707-765-4631. 9600 Petaluma, CA. USENET access

The Outer Rim, 805-252-6342. Santa Clarita, CA.

Programmer´s Exchange, 818-444-3507. El Monte, CA. Fidonet

Programmer´s Exchange, 818-579-9711. El Monte, CA.

Micro Oasis, 510-895-5985. 14.4k San Leandro, CA.

Test Engineering, 916-928-0504. Sacramento, CA.

Slut Club, 813-975-2603. USR/DS 16.8k HST/14.4K Tampa, FL. Fidonet 1:377/42

Lost City Atlantis, 904-727-9334. 14.4k Jacksonville, FL. FidoNet

Aquired Knowledge, 305-720-3669. 14.4k v.32bis Ft. Lauderdale, FL. Internet, UUCP

The Computer Mechanic, 813-544-9345. 14.4k v.32bis St. Petersburg, FL. Fidonet, Sailnet, MXBBSnet

AVSync, 404-320-6202. Atlanta, GA.

Information Overload, 404-471-1549. 19.2k ZyXEL Atlanta, GA. Fidonet 1:133/308

Atlanta Radio Club, 404-850-0546. 9600 Atlanta, GA.

Rebel BBS, 208-887-3937. 9600 Boise, ID.

Rocky Mountain HUB, 208-232-3405. 38.4k Pocatello, ID. Fionet, SLNet, CinemaNet

EchoMania, 618-233-1659. 14.4k HST Belleville, IL. Fidonet 1:2250/1, f´req LINUX

UNIX USER, 708-879-8633. 14.4k Batavia, IL. USENET, Internet mail

PBS BBS, 309-663-7675. 2400 Bloomington, IL.

Third World, 217-356-9512. 9600 v.32 IL.

Digital Underground, 812-941-9427. 14.4k v.32bis IN. USENET

The OA Southern Star, 504-885-5928. New Orleans, LA. Fidonet 1:396/1

Channel One, 617-354-8873. Boston, MA. RIME -CHANNEL

VWIS Linux Support BBS, 508-793-1570. 9600 Worcester, MA.

WayStar BBS, 508-481-7147. 14.4k V.32bis USR/HST Marlborough, MA. Fidonet 1:333/14

WayStar BBS, 508-481-7293. 14.4k V.32bis USR/HST Marlborough, MA. Fidonet 1:333/15

WayStar BBS, 508-480-8371. 9600 V.32bis or 14.4k USR/HST Marlborough, MA. Fidonet 1:333/16
Programmer´s Center, 301-596-1180. 9600 Columbia, MD. RIME
Brodmann´s Place, 301-843-5732. 14.4k Waldorf, MD. RIME -BRODMANN, Fidonet
Main Frame, 301-654-2554. 9600 Gaithersburg, MD. RIME -MAINFRAME
1 Zero Cybernet BBS, 301-589-4064. MD.
WaterDeep BBS, 410-614-2190. 9600 v.32 Baltimore, MD.
Harbor Heights BBS, 207-663-0391. 14.4k Boothbay Harbor, ME.
Part-Time BBS, 612-544-5552. 14.4k v.32bis Plymouth, MN.
The Sole Survivor, 314-846-2702. 14.4k v.32bis St. Louis, MO. WWIVnet, WWIVlink, etc
MAC´s Place, 919-891-1111. 16.8k, DS modem Dunn, NC. RIME -MAC
Digital Designs, 919-423-4216. 14.4k, 2400 Hope Mills, NC.
Flite Line, 402-421-2434. Lincoln, NE. RIME -FLITE, DS modem
Legend, 402-438-2433. Lincoln, NE. DS modem
MegaByte Mansion, 402-551-8681. 14.4 V,32bis Omaha, NE.
Mycroft QNX, 201-858-3429. 14.4k NJ.
Steve Leon´s, 201-886-8041. 14.4k Cliffside Park, NJ.
Dwight-Englewood BBS, 201-569-3543. 9600 v.42 Englewood, NJ. USENET
The Mothership Cnection, 908-940-1012. 38.4k Franklin Park, NJ.
The Laboratory, 212-927-4980. 16.8k HST, 14.4k v.32bis NY. FidoNet 1:278/707
Valhalla, 516-321-6819. 14.4k HST v.32 Babylon, NY. Fidonet (1:107/255), UseNet (die.linet.org)
Intermittent Connection, 503-344-9838. 14.4k HST v.32bis Eugene, OR. 1:152/35
Horizon Systems, 216-899-1086. USR v.32 Westlake, OH.
Horizon Systems, 216-899-1293. 2400 Westlake, OH.
Centre Programmers Unit, 814-353-0566. 14.4k V.32bis/HST Bellefonte, PA.
Allentown Technical, 215-432-5699. 9600 v.32/v.42bis Allentown, PA. WWIVNet 2578
Tactical-Operations, 814-861-7637. 14.4k V32bis/V42bis State College, PA. Fidonet 1:129/226, tacops.UUCP
North Shore BBS, 713-251-9757. Houston, TX.
The Annex, 512-575-1188. 9600 HST TX. Fidonet 1:3802/217
The Annex, 512-575-0667. 2400 TX. Fidonet 1:3802/216

Walt Fairs, 713-947-9866. Houston, TX. FidoNet 1:106/18
CyberVille, 817-249-6261. 9600 TX. FidoNet 1:130/78
splat-ooh, 512-578-2720. 14.4k Victoria, TX.
splat-ooh, 512-578-5436. 14.4k Victoria, TX.
alaree, 512-575-5554. 14.4k Victoria, TX.
Ronin BBS, 214-938-2840. 14.4 HST/DS Waxahachie (Dallas), TX. RIME, Intelec, Smartnet, etc.
VTBBS, 703-231-7498. Blacksburg, VA.
MBT, 703-953-0640. Blacksburg, VA.
NOVA, 703-323-3321. 9600 Annandale, VA. Fidonet 1:109/305
Rem-Jem, 703-503-9410. 9600 Fairfax, VA.
Enlightend, 703-370-9528. 14.4k Alexandria, VA. Fidonet 1:109/615
My UnKnown BBS, 703-690-0669. 14.4k V.32bis VA. Fidonet 1:109/370
Georgia Peach BBS, 804-727-0399. 14.4k Newport News, VA.
Top Hat BBS, 206-244-9661. 14.4k WA. Fidonet 1:343/40
victrola.sea.wa.us, 206-838-7456. 19.2k Federal Way, WA. USENET

D.2 Вне США

Galaktische Archive, 0043-2228303804. 16.8 ZYX Wien, Austria. Fidonet 2:310/77 (19:00-7:00)
Linux-Support-Oz, +61-2-418-8750. v.32bis 14.4k Sydney, NSW, Austrailia. Internet/Usenet, E-Mail/News
500cc Formula 1 BBS, +61-2-550-4317. V.32bis Sydney, NSW, Australia.
Magic BBS, 403-569-2882. 14.4k HST/Telebit/MNP Calgary, AB, Canada. Internet/Usenet
Logical Solutions, 299-9900 through 9911. 2400 AB, Canada.
Logical Solutions, 299-9912, 299-9913. 14.4k Canada.
Logical Solutions, 299-9914 through 9917. 16.8k v.32bis Canada.
V.A.L.I.S., 403-478-1281. 14.4k v.32bis Edmonton, AB, Canada. USENET
The Windsor Download, (519)-973-9330. v32bis 14.4 ON, Canada.
r-node, 416-249-5366. 2400 Toronto, ON, Canada. USENET
Synapse, 819-246-2344. 819-561-5268 Gatineau, QC, Canada. RIME-SYNAPSE
Radio Free Nyongwa, 514-524-0829. v.32bis ZyXEL Montreal, QC, Canada. USENET, Fidonet
DataComm1, +49.531.132-16. 14.4 HST Braunschweig, NDS, Germany. Fido

2:240/550, LinuxNet
DataComm2, +49.531.132-17. 14.4 HST Braunschweig, NDS, Germany. Fido 2:240/551, LinuxNet
Linux Server /Braukmann, +49.441.592-963. 16.8 ZYX Oldenburg, NDS, Germany. Fido 2:241/2012, LinuxNet
MM´s Spielebox, +49.5323.3515. 14.4 ZYX Clausthal-Zfd., NDS, Germany. Fido 2:241/3420
MM´s Spielebox, +49.5323.3516. 16.8 ZYX Clausthal-Zfd., NDS, Germany. Fido 2:241/3421
MM´s Spielebox, +49.5323.3540. 9600 Clausthal-Zfd., NDS, Germany. Fido 2:241/3422
Bit-Company / J. Bartz, +49.5323.2539. 16.8 ZYX MO Clausthal-Zfd., NDS, Germany. Fido 2:241/3430
Fractal Zone BBS /Maass, +49.721.863-066. 16.8 ZYX Karlsruhe, BW, Germany. Fido 2:241/7462
Hipposoft /M. Junius, +49.241.875-090. 14.4 HST Aachen, NRW, Germany. Fido 2:242/6, 4:30-7,8-23:30
UB-HOFF /A. Hoffmann, +49.203.584-155. 19.2 ZYX+ Duisburg, Germany. Fido 2:242/37
FORMEL-Box, +49.4191.2846. 16.8 ZYX Kaltenkirchen, SHL, Germany. Fido 2:242/329, LinuxNet (6:00-20:00)
BOX/2, +49.89.601-96-77. 16.8 ZYX Muenchen, BAY, Germany. Fido 2:246/147, info magic: LINUX (22-24,0:30-2,5-8)
Die Box Passau 2+1, +49.851.555-96. 14.4 V32b Passau, BAY, Germany. Fido 2:246/200 (8:00-3:30)
Die Box Passau Line 1, +49.851.753-789. 16.8 ZYX Passau, BAY, Germany. Fido 2:246/2000 (8:00-3:30)
Die Box Passau Line 3, +49.851.732-73. 14.4 HST Passau, BAY, Germany. Fido 2:246/202 (5:00-3:30)
Die Box Passau ISDN, +49.851.950-464. 38.4/64k V.110/X.75 Passau, BAY, Germany. Fido 2:246/201 (8:00-24:00,1:00-3:30)
Public Domain Kiste, +49.30.686-62-50. 16.8 ZYX BLN, Germany. Fido 2:2403/17
CS-Port / C. Schmidt, +49.30.491-34-18. 19.2 Z19 Berlin, BLN, Germany. Fido 2:2403/13
BigBrother / R. Gmelch, +49.30.335-63-28. 16.8 Z16 Berlin, BLN, Germany. Fido 2:2403/36.4 (16-23:00)
CRYSTAL BBS, +49.7152.240-86. 14.4 HST Leonberg, BW, Germany. Fido

2:2407/3, LinuxNet

Echoblaster BBS #1, +49.7142.213-92. HST/V32b Bietigheim, BW, Germany. Fido 2:2407/4, LinuxNet (7-19,23-01h

Echoblaster BBS #2, +49.7142.212-35. V32b Bietigheim, BW, Germany. Fido 2:2407/40, LinuxNet (20h-6h)

LinuxServer / P. Berger, +49.711.756-275. 16.8 HST Stuttgart, BW, Germany. Fido 2:2407/34, LinuxNet (8:3-17:5,19-2)

Rising Sun BBS, +49.7147.3845. 16.8 ZYX Sachsenheim, BW, Germany. Fido 2:2407/41, LinuxNet (5:30-2:30)

bakunin.north.de, +49.421.870-532. 14.4 D 2800 Bremen, HB, Germany. kraehe@bakunin.north.de

oytix.north.de, +49.421.396-57-62. ZYX HB, Germany. mike@oytix.north.de, login as gast

Fiffis Inn BBS, +49-89-5701353. 14.4-19.2 Munich, Germany. FidoNet 2:246/69,Internet,USENET,LinuxNet

The Field of Inverse Chaos, +358 0 506 1836. 14.4k v32bis/HST Helsinki, Finland. USENET; ichaos.nullnet.fi

Modula BBS, +33-1 4043 0124. HST 14.4 v.32bis Paris, France.

Modula BBS, +33-1 4530 1248. HST 14.4 V.32bis Paris, France.

STDIN BBS, +33-72375139. v.32bis Lyon, Laurent Cas, France. FidoNet 2:323/8

Le Lien, +33-72089879. HST 14.4/V32bis Lyon, Pascal Valette, France. FidoNet 2:323/5

Basil, +33-1-44670844. v.32bis Paris, Laurent Chemla, France.

Cafard Naum, +33-51701632. v.32bis Nantes, Yann Dupont, France.

DUBBS, +353-1-6789000. 19.2 ZyXEL Dublin, Ireland. Fidonet 2:263/167

Galway Online, +353-91-27454. 14.4k v32b Galway, Ireland. RIME, @iol.ie

Nemesis´ Dungeon, +353-1-324755 or 326900. 14.4k v32bis Dublin, Ireland. Fidonet 2:263/150

nonsolosoftware, +39 51 6140772. v.32bis, v.42bis Italy. Fidonet 2:332/407

nonsolosoftware, +39 51 432904. ZyXEL 19.2k Italy. Fidonet 2:332/417

Advanced Systems, +64-9-379-3365. ZyXEL 16.8k Auckland, New Zealand. Singet, INTLnet, Fidonet

Thunderball Cave, 472567018. Norway. RIME -CAVE

DownTown BBS Lelystad, +31-3200-48852. 14.4k Lelystad, Netherlands. Fido 2:512/155, UUCP

MUGNET Intl-Cistron BBS, +31-1720-42580. 38.4k Alphen a/d Rijn, Netherlands. UUCP

The Controversy, (65)560-6040. 14.4k V.32bis/HST Singapore. Fidonet 6:600/201

Pats System, +27-12-333-2049. 14.4k v.32bis/HST Pretoria, South Africa. Fidonet 5:71-1/36

Gunship BBS, +46-31-693306. 14.4k HST DS Gothenburg Sweden.

Baboon BBS, +41-62-511726. 19.2k Switzerland. Fido 2:301/580 and /581

The Purple Tentacle, +44-734-590990. HST/V32bis Reading, UK. Fidonet 2:252/305

A6 BBS, +44-582-460273. 14.4k Herts, UK. Fidonet 2:440/111

On the Beach, +444-273-600996. 14.4k/16.8k Brighton, UK. Fidonet 2:441/122

E Генеральная Общедоступная Лицензия GNU

Ниже напечатана Генеральная Общедоступная Лицензия GNU (GPL или copyleft), под которой находится Linux. Она приведена здесь, чтобы ликвидировать возникающие недоразумения относительно статуса авторских прав Linux. Это не shareware и не public domain. Большая часть ядра Linux kernel защищена copyright октябрь 1993, Linus Torvalds а другие программы и части ядра защищены copyrighted других авторов. Таким образом, Linux защищен авторскими правами, хотя вы можете распространять его при выполнении условий GPL, приведенных ниже.

ГЕНЕРАЛЬНАЯ ОБЩЕДОСТУПНАЯ ЛИЦЕНЗИЯ GNU Версия 2, Июнь 1991

Copyright (c) 1989, 1991 Free Software Foundation, Inc. 675 Mass Ave, Cambridge, MA 02139, USA. Всем разрешено копировать и распространять точные копии этого документа, содержащего лицензию, но какие-либо изменения запрещены.

E.1 Преамбула

Лицензии для большинства программ сформулированы так, чтобы лишить вас свободы в их передаче и модификации. В противоположность этому Генеральная Общедоступная Лицензия GNU стремится гарантировать вашу свободу в распространении и изменении свободнораспространяемых программ, обеспечивая их свободное распространение для всех пользователей. Эта Генеральная Общедоступная Лицензия применяется к большей части программ Free Software Foundation и ко всем прочим программам, чьи авторы разделяют эти принцыпы. (Некоторые другие программы Free Software Foundation находятся вместо этого под Библиотечной Генеральной Общедоступной Лицензией). Вы тоже можете распространить ее на свои программы.

Когда мы говорим о свободно распространяемых программах, это относится именно к свободе, а не к цене. Наши Генеральные Общедоступные Лицензии сформулированы так, чтобы гарантировать вам возможность свободно распространять копии свободнораспространяемых программ (и получать за эту работу плату, если пожелаете), что вы получите исходные тексты или сможете их получить, что вы сможете изменять программы или их части для создания новых свободнораспространяемых программ; и что вы знаете, что вы это можете делать.

Для защиты ваших прав нам необходимо наложить ограничения, которые запрещают кому бы то ни было нарушать эти права или просить вас отказаться от таких прав. Эти ограничения передают вам и определенную ответственность, если вы распространяете копии программ или модифицируете их.

Например, если вы распространяете копии таких программ бесплатно или за вознаграждение, вы должны передать получателям права на них, которые вы имели. Вы должны быть уверены, что они также получать или будут иметь возможность получить исходные коды. И вы должны сообщить эти условия, чтобы они знали свои права.

Мы защищаем ваши права в два этапа: (1) защищаем программы с помощью copyright и (2) передаем вам эту лицензию, что дает вам легальное право копировать, распространять и/или модифицировать программы.

Кроме того, для защиты каждого автора и нашей собственной защиты, мы хотим быть уверенными, что все понимают, что на свободнораспространяемые программы отсутсвует гарантия. Если программа кем-то модифицирована и передана дальше, мы хотим, чтобы получатели знали, что то, что они получили, это не оригинал, так что все проблемы, которые порождены другими, не должны бросать тень на репутацию автора оригинала.

И наконец, любой свободнораспространяемой программе постоянно угрожает патентование. Мы хотим избежать опасности, что редистрибуторы свободнораспространяемых программ самостоятнльно запатентуют программу, сделав ее своей частной собственностью. Чтобы помешать этому, мы хотим со всей определенностью сказать, что либо лицензия должна позволять всем свободно пользоваться программой, либо никому. Конкретные условия копирования, распространения и модификации приводятся ниже.

E.2 Условия копирования, распространения и модификации

0.

Эта лицензия применима к любой программе или другой работе, содержащей уведомление о владельце copyright и о том, что она может распространяться в соответсвии с требованиями Генеральной Общедоступной Лицензии. "Программа", употребляемый далее термин, относится к любой программе или работе, а также "работе основанной на Программе". Это означает, что Программа или производная работа находятся под охраной copyright: то есть работа, содержит Программу или ее часть либо дословно, либо с модификациями и/или переводами на другой язык. (Далее перевод включается без дополнительных оговорок в термин "модификация"). Далее под обращением "вы" будем понимать владельца лицензии.

Другая деятельность, отличная от копирования, распространения и модификации, не подпадает под действие этой Лицензии; она за ее пределами ее влияния. Собственно эксплуатация Программы ничем не ограничивается и выходные данные, выдаваемые Программой попадают под Лицензию, только если их можно отнести к "работе, основанной на Программе" (независимо от того, что данные получены в результате выполнения Программы). Либо действительно зависит от того, что делает Программа.

1.

Вы можете копировать и распространять точные копии исходных текстов Программы, как вы их получили, на любом носителе, снабженном заметным и подобающим образом представленным уведомлением о copyright и уведомлением об отсутствии гарантии; сохраняйте без изменений записи, относящиеся к этой Лицензии и отсутствию гарантии и передавайте другим вместе с копией Программы копию этой Лицензии.

Вы можете получать плату за саму физическую процедуру передачи копии и вы можете за плату под свою ответсвенность предложить (от своего имени) гарантии.

2.

Вы можете модифицировать свою копию Программы или ее части, таким образом реализуя работы, основанные на Программе, а далее копировать и распределять эти модификации и другие работы в терминах Раздела 1, обеспечив выполнение всех условий:

a.

Модифицированные файлы вы должны снабдить на видном месте в начале уведомлением, что вы изменили файлы, и датой изменений.

b.

Вы должны снабдить любую работу, которую вы распространяете или публикуете, как целиком, так и частями, или полученную на основании Программы или ее части, лицензией, позволяющей бесплатно передавать третим лицам на основании Лицензии.

c.

Если модифицированная программа должна читать в интерактивном режиме команды в процессе выполнения, вы должны обеспечить, чтобы в начале работы она автоматически выдавала об'явления, включющие соответсвующий copyright и уведомление об отсутствии гарантии (либо, наоборот, сообщение, что вы предоставляете гарантию) и что пользователь может заниматься дальнейшим распространением программы при соблюдении этих условий, а также информацию пользователю, где можно увидеть копию этой лицензии. (Исключение: Если Программа сама по себе интерактивная, но в норамльном режиме не выдает такие сообщения, ваша работа, базирующаяся на Программе не обязана выдавать эти об'явления).

Эти требованияприменимы к работам, связанным с модификацией вцелом. Если выделенная часть этой работы не является производной от Программы, и может обоснованно рассматриваться как независимая и отдельная работа, тогда эта Лицензия и ее составляющие не применимы к этим частям, когда они распространяются как независимые работы. Но когда вы распространяете эти части внутри целого, что можно определить, как работу на основе Программы, распространение целого должно попадать под требования этой Лицензии, чьи ограничения для получателей лицензии будут распространяться на целое, и таким образом все части, вне зависимости от того, кто что написал.

Таким образом, это не следует воспринимать как попытку заявить свои права или оспаривать ваши права, на работу написанную целиком вами, а намерение реализовать право управления распространением производных или коллективных работ , основанных на Программе.

В дополнение, простое об'единение другой работы, не основанной на Программе с Программой (или работой, основанной на Программе) на одном томе хранения или средстве распространения не переносит действие Лицензии на (эту) другую работу.

3.

Вы можете копировать и распространять Программу (или работу, основанную на программе из Раздела 2) в об'ектных кодах или выполняемой форме с учетом требований Разделов 1 и 2, при условии, что вы также выполните одно из следующего:

a.

Сопроводите ее полными машинно-читаемыми исходными текстами, которые должны распространяться с учетом требований Разделов 1 и 2, на средстве кастомизации, используемом для обмена программами; или

b.

Сопроводите ее письменным предложением, действительным не менее, чем в течение трех лет, продать любой третьей стороне по стоимости, не большей, чем стоимость физического копирования дистрибутива, полную машинно-читаемую копию соответсвующего исходного кода, который можно распространять с учетом ограничений Разделов 1 и 2, на средствах кастомизации, используемых для обмена программами; или

c.

Сопроводить ее информацией, которую вы получили в качестве предложения по распространению соответсвующего исходного кода. (Эта альтернатива допускается только для некоммерческого распространения, и только если вы получили программу в качестве предложения в об'ектных кодах или выполняемой форме, в соответствии с Подразделом b, приведенным выше).

Исходный код предпочтителен для работ по модификации. Полный исходный код подразумевает весь исходный код для содержащихся модулей, плюс все файлы определения интерфейса, плюс сценарии управления компиляцией и инсталляцией. Однако, в качестве специфического исключения, распространяемый исходный код не обязан содержать то, что обычно распространяется (в исходных текстах или бинарном виде) с основными программными компонентами (компилятором, ядром и т.п.) операционной системы, на которой программа выполняется, за исключением случая, когда сама компонента сопровождает выполняемые файлы.

Если распространение выполняемых или об'ектных кодов реализуется предоставлением доступа для копирования из указанного места, тогда предоставление эквивалентного доступа для копирования исходных исходного кода из того же места считается распространением исходного кода, даже если третьи стороны не принуждаются копировать исходный код вместе с об'ектным кодом.

4.

Вы не должны копировть, модифицировать, дополнительно лицензировать или распространять Программу иначе, чем под данной Лицензией. Иначе любая попытка копировть, модифицировать, дополнительно лицензировать или распространять Программу будет нарушением и автоматически аннулирует ваши права, защищаемые данной Лицензией. Однако третьих сторон, получившие копии или права по этой Лицензии, они не аннулируются, пока они их не нарушают.

5.

Вы можете не следовать этой Лицензии, коль скоро вы ее не подписывали. Однако, ничто другое не дает вам права модифицировать или распространять Программу или производные из нее работы. Эти действия запрещены законом, если вы не принимаете требования Лицензии. Поэтому, занимаясь модификацией или распространением Программы (или любой работы, основанной на Программе) вы должны продемонстрировать вашу приверженность этой Лицензии, в связи с этими работами, и всем требованиями и условиям по копированию, распространению или модификации Программы или работ, основанных на Программе.

6.

Каждый раз, когда вы передаете Программу (или работу, основанную на Программе), получатель автоматически получает лицензию от первого владельца лицензии на копирование, распространение или модификацию Программы с учетом требований и условий. Вы не имеете права накладывать дополнительные ограничения на использование Программы теми, кому вы ее передаете. Вы не отвечаете по претензиям, пред'являемым третьими сторонами к этой Лицензии.

7.

Если по решению суда или по заявлению о нарушении патента или по другой причине (не ограниченной патентными спорами) на вас накладываются требования (решением суда, действующим договором или чем-то иным), которые противоречат условиям Лицензии, это не является основанием для нарушения вами условий Лицензии. Если вы не можете одновременно выполнять условия Лицензии и выполнять другие обязательства, то вы вообще не имеете права распространять Программу. Например, если требования патента не разрешают бесприбыльное распространение Программы всеми теми, кто прямо или опосредованно получил от вас копии, тогда единственный вариант, позволяющий не нарушать как те, так и другие требования, заключается в отказе от распространения Программы вообще.

Если какая-то часть этого раздела не применима к конкретной ситуации или невыполнима при некоторых конкретных обстоятельствах, то предполагается, что часть раздела или весь раздел применимы в других обстоятельствах.

Целью этого раздела не является попытка подтолкнуть вас к нарушению каких-либо патентных прав или прав собственности, или оспаривать справедливость таких прав. Этот раздел имеет единственную цель - защиту целостности системы свободно распространяемых программ, которая реализуется практикой общедоступных лицензий. Многие люди внесли большой вклад в обширный набор свободно распространяяемых программ, полагаясь на эту систему. Это право автора/дарителя решать, желает ли он (она) распространять программы через какие-то другие системы и мы не может навязывать выбор.

Целью данного раздела было уточнение того, что является следствием последующего содержания Лицензии.

8.

Если распространение и/или использование Программы ограничено в некоторых странах либо патентами, либо запатентованными интерфейсами, исходный владелец патента, который поместил Программу по данную Лицензию, может добавить явно указанную географию распространения, исключив соответсвующие страны, так что распространение будет разрешено среди стран, не попавших в список исключенных. В этом случае данная Лицензия включает ограничения, как неот'емлемую часть своего текста.

9.

The Free Software Foundation (FSF) может публиковать и/или изменять время от времени Генеральную Общедоступную Лицензию. Эти новые версии будут аналогичны по духу данной версии, но могут отличаться в деталях, связанных с вновь возникшими проблемами.

Каждая версия имеет уникальный номер. Если Программа указывает номер версии данной Лицензии, которая включает и ”любую более раннюю версию”, вы можете следовать требованиям либо данной версии, либо любой более ранней версией, опубликованной Free Software Foundation. Если Программа не указывает номер версии Лицензии, вы можете выбрать любую когда-либо опубликованную Free Software Foundation версию.

10.

Если вы хотитевставить части Программы в другие свободно распространяемые программы, чьи условия распространения отличаются, напишите автору и спросите его разрешения. Для программ, защищенных copyright Free Software Foundation, напишите в Free Software Foundation; мы иногда делаем для этого исключения. Наше решение следует двум целям сохранения статуса свободно распространяемых программ для всех производных из этих программ, и вообще, поощрение распространения программ.

ОТКАЗ В ГАРАНТИИ

11.

ПОСКОЛЬКУ
ПРОГРАММА ИМЕЕТ ЛИЦЕНЗИЮ БЕСПЛАТНО РАСПРОСТРАНЯЕМОЙ ПРОГРАММЫ, НА ПРОГРАММУ ОТСУТСТВУЕТ ГАРАНТИЯ (В ПРЕДЕЛАХ ДОПУСКАЕМЫХ ЗАКОНОМ). ЗА ИСКЛЮЧЕНИЕМ ТЕХ СЛУЧАЕВ, КОГДА ПИСЬМЕННО УКАЗАНО ОБРАТНОЕ, ВЛАДЕЛЬЦЫ COPYRIGHT И/ИЛИ ДРУГИЕ ПОСТАВЛЮТ ПРОГРАММУ ”КАК ЕСТЬ” БЕЗ КАКОЙ БЫ ТО НИ БЫЛО ГАРАНТИИ, ЯВНОЙ ИЛИ ПОДРАЗУМЕВАЕМОЙ, ВКЛЮЧАЯ, НО НЕ ОГРАНИЧИВАЯ ПОДРАЗУМЕВАЕМЫЕ ГАРАНТИИ, НЕОБХОДИМЫЕ ПРИ ТОРГОВЛЕ И ОБЯЗАТЕЛЬНЫЕ ДЛЯ ДРУГИХ КОНКРЕТНЫХ ЦЕЛЕЙ. ВЫ БЕРЕТЕ НА СЕБЯ ВЕСЬ РИСК ЗА КАЧЕСТВО И РЕЗУЛЬТАТЫ РАБОТЫ ПРОГРММЫ. ЕСЛИ ПРОГРАММА ОКАЖЕТСЯ НЕИСПРАВНОЙ, ВЫ НЕСЕТЕ ВСЕ РАСХОДЫ ПО НЕОБХОДИМОМУ ОБСЛУЖИВАНИЮ, РЕМОНТУ ИЛИ КОРРЕКТИРОВКЕ.

12.

НИ В КАКИХ СЛУЧАЯХ, КРОМЕ СЛЕДУЮЩИХ ИЗ ЗАКОНОВ ПО ИСПОЛЬЗОВАНИЮ ИЛИ СОГЛАСИЯ В ПИСЬМЕННОЙ ФОРМЕ, НЕ БУДЕТ НИ ОДИН ВЛАДЕЛЕЦ COPYRIGHT ИЛИ ДРУГОЕ ЛИЦО, КОТОРОЕ МОГЛО МОДИФИЦИРОВАТЬ ПРОГРАММУ И/ИЛИ ДАЛЕЕ РАСПРОСТРАНЯТЬ ЕЕ В СООТВЕТСТВИИ С УСТАНОВЛЕННЫМИ РАЗРЕШЕНИЯМИ, НЕ БУДЕТ НЕСТИ ОТВЕТСТВЕННОСТЬ ЗА НАНЕСЕННЫЙ ПРОГРАММОЙ УЩЕРБ, ВКЛЮЧАЯ ЛЮБОЙ УЩЕРБ ОБЩЕГО ХАРАКТЕРА, СПЕЦИАЛЬНЫЙ, СЛУЧАЙНЫЙ ИЛИ ЯВЛЯЮЩИЙСЯ СЛЕДСТВИЕМ ПОВРЕЖДЕНИЯ, ВЫЯВЛЕННОГО ПРИ ИСПОЛЬЗОВАНИИ ПРОГРАММЫ ИЛИ НЕВОЗМОЖНОСТИ ИСПОЛЬЗОВАТЬ ПРОГРАММУ (ВКЛЮЧАЯ, НО НЕ ОГРАНИЧИВАЯСЬ, ПОТЕРЮ ДАННЫХ ИЛИ ИХ НЕПРАВИЛЬНУЮ ПЕРЕДАЧУ
ИЛИ ПОТЕРИ, КОТОРЫЕ ПОНЕСЛИ ВЫ ИЛИ ТРЕТЬЯ СТОРОНА, ИЛИ НЕСПОСОБНОСТЬ ПРОГРАММЫ ВЗАИМОДЕЙСТВОВАТЬ С ДРУГИМИ ПРОГРАММАМИ), ДАЖЕ ЕСЛИ У ВЛАДЕЛЬЦЕВ ИЛИ ТРЕТЬЕЙ СТОРОНЫ КОНСУЛЬТИРОВАЛИСЬ ПО ПОВОДУ ВОЗМОЖНОСТИ ТАКОГО УЩЕРБА.

КОНЕЦ ТРЕБОВАНИЙ И УСЛОВИЙ

E.3 Как Применять Эти Требования к Вашим Новым Программам

Если вы создали новую программу и хотите принести максимальную возможную пользу человечеству, лучше всего этого достигнуть, сделав ее свободнораспространяемой, что позволит всем и каждому распространять и изменять ее при выполнении этих требований.

Чтобы это сделать, добавьте следуюшие записи к своей программе. Самое надежное, это добавить их в начало кажого файла исходных текстов, для максимального эффекта добавив туда же информацию об отсутствии гарантии; каждый файл должен как минимум содержать строку “copyright” и указание, где есть полный текст лицензии.

```
одна строка, чтобы привести имя программы и краткий смысл того,
 что она делает Copyright (c) 19yy имя автора

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

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

   You should  have received  a copy  of the  GNU General  Public
   License along with  this program;  if not,  write to the  Free
   Software Foundation, Inc., 675 Mass Ave, Cambridge,  MA 02139,
   USA.
```

Добавьте также информацию, как с вами связаться по электронной и обычной почте. Если программа интерактивная, сделайте небольшую выдачу на экран в начале работы программы, вроде этой:

```
   Gnomovision version  69,  Copyright (C)  19yy name  of  author
   Gnomovision comes  with ABSOLUTELY  NO WARRANTY;  for  details
   type `show w´.   This  is free software,  and you are  welcome
   to redistribute it under certain conditions; type `show c´ for
   details.
```

Гипотетические команды ‘show w’ и ‘show c’ должны показать соответсвующие части the General Public License. Разумеется, команды, которые будете использовать вы, могут называться как-то иначе, чем ‘show w’ и ‘show c’; их даже можно реализовать мышкой или через пункты меню - как больше подходит вашей программе.

Вы должны также поговорить со своим работодателем (если вы работаете в качестве программиста) или в школе, если вы в ней, чтобы подписать "отказ от авторских прав" (''copyright disclaimer'') на программу, если это необходимо. Вот образец:

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

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

55

Эта Генеральная Общедоступная Лицензия (GPL) не разрешает вставлять вашу программу другие программы, представляющие чью-то частную собственность Если ваша программа - лишь одна из программ библиотеки, может быть более целесообразно будет разрешить включать (link) ее в любые прикладные программы в качестве библиотечной функции Если этого достаточно, используйте Библиотечную Генеральную Общедоступную Лицензию GNU (GNU Library General Public License) вместо Лицензии GPL.

56

F The GNU General Public License

57

Printed below is the GNU General Public License (the GPL or copyleft), under which Linux is licensed. It is reproduced here to clear up some of the confusion about Linux's copyright status---Linux is not shareware, and it is not in the public domain. The bulk of the Linux kernel is Copyright (c) 1993 by Linus Torvalds, and other software and parts of the kernel are copyrighted by their authors. Thus, Linux is copyrighted, however, you may redistribute it under the terms of the GPL printed below.

GNU GENERAL PUBLIC LICENSE Version 2, June 1991

---

[55] (прим. переводчика: Есть подозрение, что очень (и даже абсолютно) желательно писать это на английком языке).

[56] (прим.переводчика: Юридический слог, которым написана Лицензия, допускает лишь "литературный" перевод. Для точной передачи смысла надо иметь достаточный юридический опыт. Поэтому, пользуясь случаем и требованиями Лицензии, хочу уведомить, что несмотря на принятые мучения, не могу гарантировать точность перевода. Оригинал Лицензии прилагается).

[57] Примечание переводчика. Чтобы избежать недоразумений, связанных с переводом юридического документа на русский язык, ниже мы приводим оригинальный текст лицензии GNU.

F.1 Preamble

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

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

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

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

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

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

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

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

F.2 Terms and Conditions for Copying, Distribution, and Modification

0.

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

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1.

You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2.

You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a.

You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b.

You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c.

If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

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

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program. In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3.

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

a.

Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b.

Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c.

Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

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

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4.

You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5.

You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6.

Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7.

If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

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

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

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8.

If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9.

The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10.

If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11.

BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED

ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

F.3 How to Apply These Terms to Your New Programs

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

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

```
one line to give the program´s name and  a brief idea of what
it does. Copyright c019yy name of author

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

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

You should  have received  a copy  of the  GNU General  Public
License along with  this program;  if not,  write to the  Free
Software Foundation, Inc., 675 Mass Ave, Cambridge,  MA 02139,
USA.
```

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

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

```
Gnomovision version 69, Copyright (C) 19yy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details
type `show w´. This is free software, and you are welcome
to redistribute it under certain conditions; type `show c´ for
details.
```

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

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

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

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

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