Денис Колисниченко

АДМИНИСТРИРОВАНИЕ UNIX-СЕРВЕРА и LINUX-СТАНЦИЙ

Операционные системы Linux и FreeBSD
Миграция Windows-сервера и рабочих станций на UNIX
Резервное копирование и восстановление системы
Подсчет и мониторинг трафика

Денис Колисниченко

АДМИНИСТРИРОВАНИЕ UNIX-СЕРВЕРА и LINUX-СТАНЦИЙ

ПИТЕР®

Москва · Санкт-Петербург · Нижний Новгород · Воронеж
Ростов-на-Дону · Екатеринбург · Самара · Новосибирск
Киев · Харьков · Минск
2011

ББК 32.988.02
УДК 004.738.5
К60

Колисниченко Д. Н.
К60 Администрирование Unix-сервера и Linux-станций. — СПб.: Питер, 2011. — 400 с.: ил.

ISBN 978-5-459-00748-0

Книга описывает процесс развертывания и администрирования сети на основе Unix-сервера и Linux-станций. Автор предлагает готовые решения для быстрой установки и настройки локальной сети. Большое количество примеров и готовых настроек позволяет использовать эту книгу в качестве практического руководства для работы.

Издание предназначено для системных администраторов и опытных пользователей.

ББК 32.988.02
УДК 004.738.5

ISBN 978-5-459-00748-0

Краткое содержание

Содержание

Введение

Давно прошли те времена, когда UNIX-система считалась чем-то экзотическим, а специалистов по ней днем с огнем не найти. А Linux давно «поселилась» на компьютерах домашних и мобильных пользователей — ноутбук с предустановленным дистрибутивом Linux можно купить в любом магазине.

Раньше приходилось долго объяснять, почему вместо привычной версии Windows на компьютер установлена неизвестная операционная система Linux. Сейчас ответ знают, если не все, то многие: за Linux не нужно платить. И именно эта операционная система делает конечный продукт (будь-то настольный компьютер, ноутбук или устройство со встроенной операционной системой, например маршрутизатор) дешевле. Сравните цены на аналогичные модели компьютеров, но с предустановленной операционной системой Windows они будут дороже.

В этой книге вы узнаете, как сэкономить на программном обеспечении в масштабах предприятия. Да, мы постараемся установить бесплатную и свободно распространяемую операционную систему на каждый компьютер вашей фирмы — от сервера до компьютера секретаря. Сразу хочу отметить, что перевести абсолютно все компьютеры на бесплатное программное обеспечение не получится. Некоторые Windows-программы попросту отказываются запускаться в эмуляторе Windows, а их аналогов еще не существует в природе либо они недостаточно функциональны. Поэтому некоторые компьютеры все же придется оставить под управлением Windows.

Сколько мы сэкономим?

Давайте подсчитаем, сколько получится сэкономить, отказавшись от проприетарного (коммерческого) программного обеспечения. Пусть у нас есть среднее предприятие, использующее 20 рабочих станций и один сервер, выполняющий все основные функции — и шлюз, и сервер баз данных, и прокси. Часто так и бывает, хотя правильнее купить несколько серверов и распределить их задачи.

Раньше, когда планировалась стоимость компьютерного парка, не учитывалась стоимость программного обеспечения. А зачем? Ведь за пару долларов можно

купить на рынке диск со всеми необходимыми программами — на нем будет и Windows Server, и настольная версия Windows и пакет MS Office. Сейчас все сложнее, так как использование нелицензионного ПО преследуется по закону.

По минимуму, нам нужно купить 20 копий Windows 7, 20 копий пакета MS Office и одну копию Windows Server 2008. Одна копия Windows 7 Pro (а именно такую имеет смысл использовать в корпоративной среде) обойдется примерно в 220 долларов. Умножаем это число на 20 и получаем 4400 американских долларов. На офисном пакете можно сэкономить, купив MS Office Standard 2010 за 380 долларов. На 20 компьютеров — это 7740 долларов. Осталось купить Windows Server 2008, стандартная редакция которого обойдется в 750 долларов. Подсчитаем: 4400 + 7740 + 750 = 8930 долларов.

Почему так дорого? Можно сэкономить, купив OEM-версии Windows — они существенно дороже, но OEM-версии продаются исключительно с новыми компьютерами. А сейчас речь идет о миграции уже существующего предприятия либо на лицензионное, либо на свободное программное обеспечение. Поэтому при расчете учитывалась стоимость коробочных версий Windows. Сэкономить на MS Office тоже не получится. Да, версия для студентов стоит всего 150 долларов, но предприятие никак не относится к учебному заведению, поэтому использование лицензионного MS Office Home and Student будет приравниваться к нелицензионной копии, поскольку нарушены условия лицензии.

Первое, что приходит в голову, — это сэкономить на пакете MS Office. Ведь есть же бесплатный офисный пакет — OpenOffice.org, который может работать как в Linux, так и в Windows. Но зачем тогда нужна сама Windows, если мы отказались от MS Office? Можно пойти дальше и сэкономить еще больше, установив Linux. А заключительный этап перехода на свободное ПО — это установка бесплатной ОС на наш сервер.

Какие операционные системы будут рассмотрены в книге

Особой популярностью из операционных систем семейства UNIX пользуются только две: FreeBSD и Linux. Linux — более универсальна и ее можно использовать как на сервере, так и на рабочей станции. FreeBSD больше подходит, все же, только для сервера. Конечно, при особом желании можно превратить FreeBSD в рабочую станцию, но это потребует гораздо больше усилий, чем в случае с Linux, где все, что нужно сделать, — это просто установить необходимые пакеты.

Читатели, знакомые с Linux, знают, что есть различные дистрибутивы этой операционной системы, созданные различными разработчиками. В этой книге не будет привязки к какому-нибудь конкретному дистрибутиву. Вместо этого будут описаны особенности настройки популярных дистрибутивов Linux. По возможности я буду стараться больше привязываться к конфигурационным файлам (особенно при настройке сервера), чем к программам-конфигураторам, чтобы потом не переписывать всю книгу, если разработчики дистрибутива надумают изменить название какой-то кнопки в конфигураторе.

Как правильно читать эту книгу

Предположим, что вы выбрали обе операционные системы: FreeBSD для сервера и Linux для рабочей станции. Тогда читайте все главы подряд. Сначала мы установим операционные системы, затем научимся администрировать их, после этого сначала переведем все рабочие станции предприятия на Linux, а уж после займемся миграцией сервера на FreeBSD.

Но, может, вы уже немного знакомы с Linux и, чтобы упростить себе жизнь, решите устанавливать Linux на абсолютно все компьютеры и сервер в том числе. В этом случае читайте только те пункты глав, которые относятся к настройке Linux, и смело пропускайте пункты или даже целые главы, где описывается настройка FreeBSD.

Прежде чем вы приступите к чтению этой книги, скажу, чего точно в ней не будет. Как вы уже успели понять, книга посвящена миграции сети предприятия на UNIX. Если вы на данный момент не являетесь администратором и не имеете малейшего понятия о том, что такое маска сети, класс сети, IP-адрес сети (то есть не знаете основ TCP/IP), вам понадобится как минимум еще одна книга, где все эти основы описываются. Таких книг предостаточно, поэтому у вас не будет проблем с ее покупкой.

Приятного чтения книги!

От издательства

Ваши замечания, предложения и вопросы отправляйте по адресу электронной почты comp@piter.com (издательство «Питер», компьютерная редакция). Мы будем рады узнать ваше мнение! Подробную информацию о наших книгах вы найдете на веб-сайте издательства www.piter.com.

I

Знакомство с системой

Установка Linux

1

1.1. Системные требования

Системные требования современных дистрибутивов Linux меня огорчают. Раньше для Linux вполне достаточно было 256 Мбайт оперативной памяти, теперь о графическом режиме можно забыть, если у вас менее 384 Мбайт «оперативки». Даже если у вас 384 Мбайт, но встроенная видеокарта, которая «черпает» ресурсы оперативной памяти, то «графика» в Linux не запустится и даже установка будет в текстовом режиме. Модули оперативной памяти имеют объем, равный степени двойки: 64, 128, 256, 512, 1024 и т. д. 384 Мбайт — это один модуль 256 Мбайт и один модуль 128 Мбайт. Два модуля по 256 Мбайт — это 512 Мбайт. Отсюда можно сделать вывод, что минимальный объем оперативной памяти для современного дистрибутива Linux — 512 Мбайт.

Но ведь и для Windows 7 минимальный объем ОЗУ — 512 Мбайт. А Linux всегда отличалась своей меньшей прожорливостью. Единственное, что радует, так это то, что Linux будет быстрее работать на компьютере с 512 Мбайт оперативной памяти, чем Windows 7. Требование к жесткому диску тоже огорчает. Редко какой дистрибутив сейчас требует 3 Гбайт. Да, некоторые дистрибутивы могут похвастаться меньшим минимальным размером винчестера, но, как правило, это минимальные установки, подходящие разве что для организации шлюза, но никак не для полноценного использования системы в качестве рабочей станции. Но 3 Гбайт — это самый минимум, которого едва хватит на установку необходимых вам программ. Желательно все-таки не пожадничать и для программ зарезервировать 4 Гбайт. Потом нужно еще место под ваши пользовательские данные. Хотя бы под музыку, поскольку стандартный фильм в формате AVI в хорошем качестве занимает 1,4 Гбайт. Добавим еще 2 Гбайт для пользовательских файлов. Выходит 6 Гбайт, и учитывая небольшой объем оперативной памяти, нужно отвести 512 Мбайт под раздел подкачки. Поэтому самый минимум для установки Linux — 6,5 Гбайт. Опять получились требования к жесткому диску на уровне Windows 7. Хочу заметить, что все указанные требования не относятся к какому-нибудь конкретному дистрибутиву, а к любому абстрактному и современному дистрибутиву Linux. Так что рассматривайте данные требования как практически

рекомендации — ведь наша цель создать рабочую станцию, а не просто установить Linux и гордо ударить себя по груди[1].

Почему я так много внимания уделяю системным требованиям? Ведь на любом современном компьютере как минимум 1 Гбайт оперативной памяти и жесткий диск на 160–250 Гбайт. Да, это так. Любой современный дистрибутив можно установить на любой современный компьютер. Но раньше конек Linux был в возможности организации полноценного рабочего места на старых компьютерах, где Windows тормозила. Так вот, «планка» требования к «железу» существенно выросла. Скажем так, если три года назад можно было взять совсем древний компьютер на уровне Pentium-II или Celeron (или даже просто Pentium) с 128–256 Мбайт оперативной памяти, жестким диском 2–4 Гбайт и установить на него Linux (причем не просто Linux, а самый современный дистрибутив), то сейчас нужно искать более мощный компьютер. К тому же сейчас очень популярны нетбуки — небольшие ноутбуки без привода DVD. Но на таких компьютерах емкость жесткого диска, как правило, ограничена. Нормальный объем SSD-накопителя нетбука (версии нетбуков с полноценными жесткими дисками не рассматриваем) — от 4 до 20 Гбайт. Чего будет маловато для установки дистрибутива. Поэтому если вы купили самый скромный нетбук, вам нужно искать и соответствующий дистрибутив. Можете обратить свое внимание на Ubuntu Netbook Remix — современный дистрибутив, адаптированный для использования на ноутбуках. Системные требования умеренные, а программное обеспечение адаптировано под использование на небольшом дисплее нетбука. Ознакомиться с Ubuntu Netbook Remix можно по адресу http://www.ubuntu.com/netbook/

1.2. Подготовка к установке

Linux не может использовать Windows-разделы в качестве корневой файловой системы, то есть невозможно установить Linux непосредственно на Windows-раздел. В процессе установки нам нужно будет уменьшить размер одного из Windows-разделов, а на освободившемся месте создать Linux-разделы.

До установки Linux нужно дефрагментировать Windows-раздел, размер которого вы собрались уменьшать. Выяснить, какой раздел нужно уменьшить, просто: на этом разделе должно быть достаточно свободного места (как мы выяснили, минимум 6,5 Гбайт).

Запустите дефрагментатор Windows (Пуск ▸ Стандартные ▸ Служебные ▸ Дефрагментация диска или Пуск ▸ Все программы ▸ Стандартные ▸ Служебные ▸ Дефрагментация диска — в Windows Vista/7). Дефрагментация необходима, чтобы все данные были перемещены в «начало» раздела, без этого невозможно выполнить изменение размера раздела без потери данных.

[1] Строго говоря, если четко задать цели функционирования конкретной рабочей станции и заранее определиться с используемым на ней программным обеспечением, то возможно использование и других дистрибутивов, с более скромными системными требованиями как к объему оперативной памяти, так и объему жесткого диска, измеряемыми всего лишь десятками мегабайт. И эти дистрибутивы тоже будут вполне соответствовать абстракции «современный дистрибутив Linux». — *Здесь и далее примеч. науч. ред.*

1.3. Установка Linux

Хотя в книге рассматривается несколько дистрибутивов Linux, рассмотреть установку каждого дистрибутива мы не сможем физически. У каждого дистрибутива своя программа установки со своими особенностями. Но в общих чертах процесс установки подобен: выбор языка системы, выбор раскладки клавиатуры, разметка жесткого диска, выбор программного обеспечения для установки, установка программного обеспечения и загрузчика Linux, установка пароля администратора, добавление пользователей, базовая настройка системы (сеть, видео, звук и т. д.) и перезагрузка. Так в общих чертах можно описать процесс установки абстрактного дистрибутива. А в этой главе мы рассмотрим установку дистрибутива Mandriva 2010.1 Spring — самой последней версии Linux Mandriva. Программа установки Mandriva классическая — после установки вы получаете практически настроенную систему. У других дистрибутивов бывают и упрощенные программы установки, например в дистрибутиве Ubuntu фактически производится только разметка жесткого диска и перенос образа системы на созданный раздел. Вся настройка осуществляется после установки системы. К тому же Mandriva обладает самой удачной программой разметки жесткого диска. И Mandriva 2010.1 Spring может быть установлена на компьютер с 256 Мбайт оперативной памяти в графическом режиме, в отличие от других дистрибутивов! Все иллюстрации для этой главы были сделаны на компьютере с 256 Мбайт оперативной памяти — эксперимента ради.

Рис. 1.1. Загрузочное меню инсталляционного диска

Рис. 1.2. Выбор языка в загрузочном меню

Рис. 1.3. Загрузочное меню на русском языке

1.3.1. Запуск программы установки Linux

Загрузитесь с установочного DVD. Вы увидите загрузочное меню инсталляционного диска (рис. 1.1).

Можно сразу выбрать команду Install Linux Mandriva 2010 Spring и нажать Enter, а можно нажать F2 для выбора языка (тогда весь дальнейший процесс установки будет произведен на выбранном вами языке). Как видите, после выбора русского языка (рис. 1.2) команды загрузочного меню сразу были русифицированы (рис. 1.3)

Если нажать F3, вы сможете отредактировать параметры загрузки. Обычно их редактировать не нужно, но иногда установка Linux отказывается запускаться, а пользователь видит сообщение

```
kernel panic – not syncing: IO-APIC + timer doesn't work!
```

В этом случае нужно добавить параметр `noapic` в список параметров (рис. 1.4). Также полезен параметр `edd=off`, если параметры жесткого диска (например, размеры разделов) определены неверно. Как правило, с Mandriva таких проблем не возникает, зато некоторым версиям openSUSE полезно передать параметр `edd=off`, особенно при установке их на редкие («экзотические») виды жестких дисков.

Рис. 1.4. Редактирование параметров загрузки

1.3.2. Выбор языка

Если вы ранее поленились выбрать русский язык, то окно, изображенное на рис. 1.5, будет представлено на английском языке. Что нужно сделать, думаю понятно: нужно выбрать русский язык.

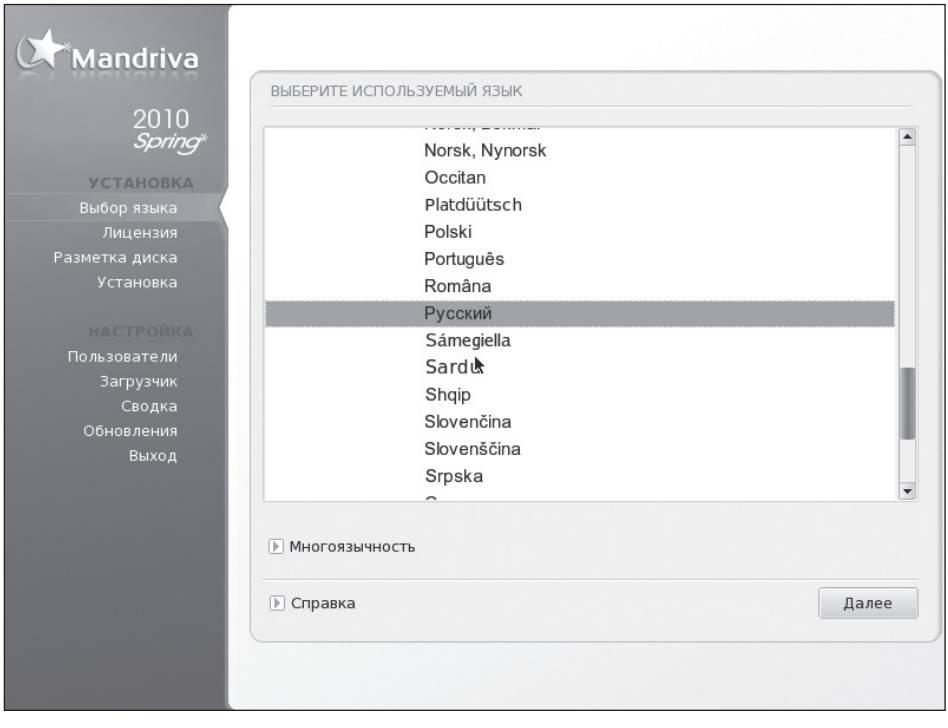

Рис. 1.5. Выбор языка

1.3.3. Лицензионное соглашение

Читать лицензионное соглашение или нет, личное дело каждого (рис. 1.6). Но для продолжения установки вы обязаны принять лицензионное соглашение.

1.3.4. Выбор раскладки клавиатуры

После прочтения лицензии нужно выбрать раскладку клавиатуры. Особо выбирать не из чего — выбираем русскую раскладку и нажимаем Далее (рис. 1.7). Кнопку Больше можно не нажимать — вы увидите список дополнительных раскладок, которые, как правило, используются очень редко.

Далее нужно выбрать комбинацию клавиш, использующуюся для переключения раскладок. Не знаю как вам, а мне больше всего нравится комбинация Ctrl + Shift (рис. 1.8).

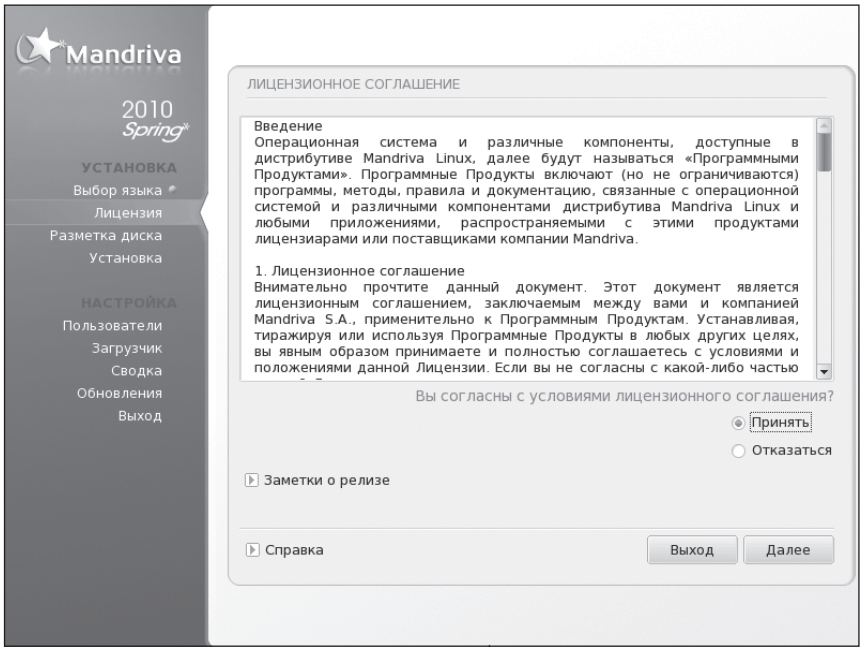

Рис. 1.6. Лицензионное соглашение

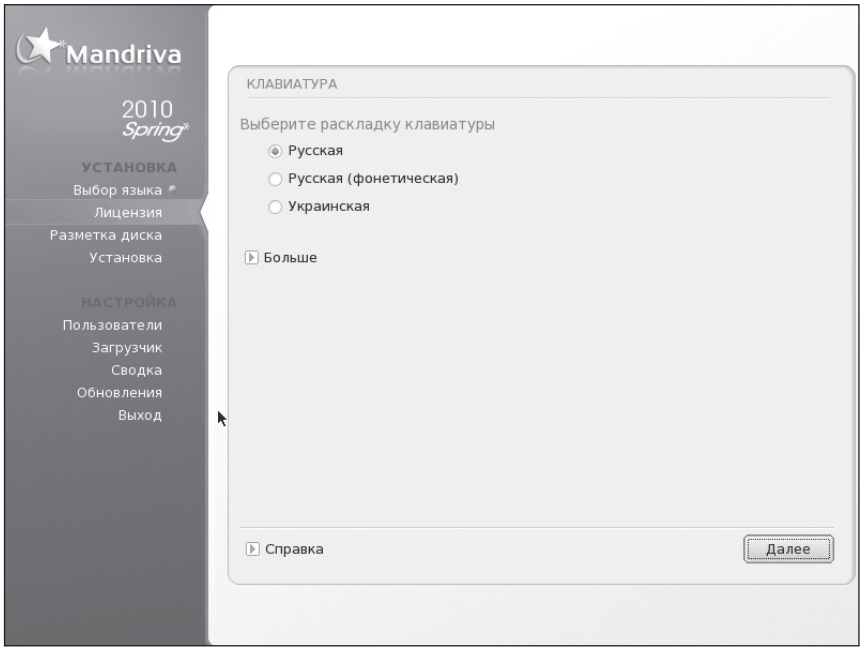

Рис. 1.7. Выбор раскладки клавиатуры

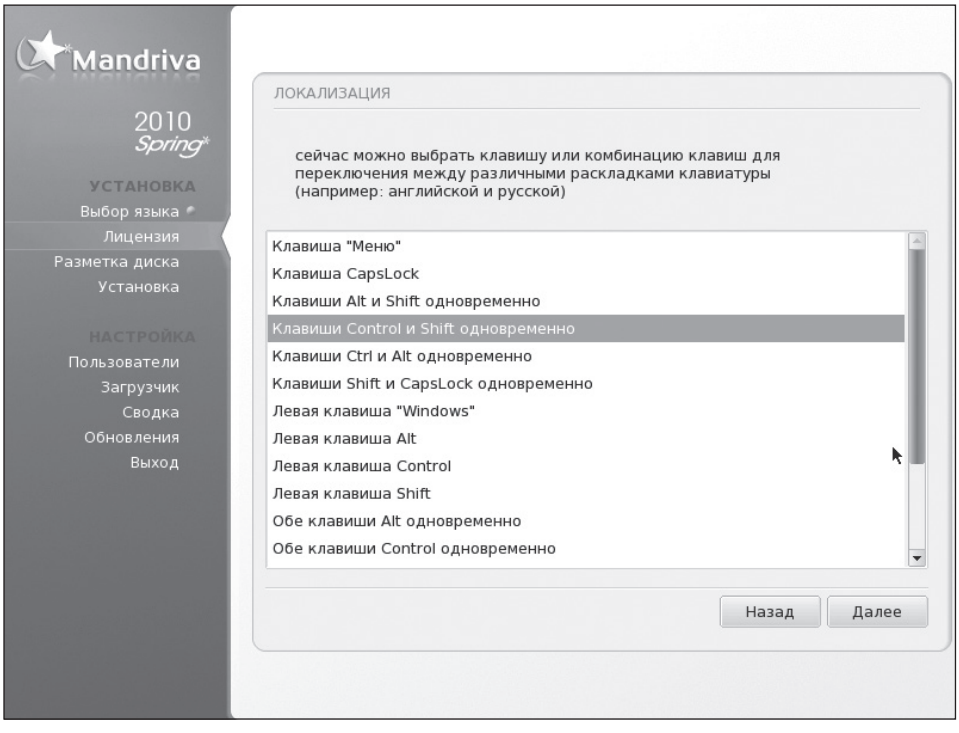

Рис. 1.8. Выбор способа переключения раскладок клавиатуры

Выбранная вами комбинация клавиш начнет действовать после перезагрузки системы, а в процессе установки для переключения раскладок можете использовать правую клавишу Control.

1.3.5. Создание разделов Linux

Наступил самый ответственный момент — создание разделов Linux. Программа установки спросит вас, как вы хотите произвести разметку диска: автоматически или вручную (рис. 1.9). Нужно выбрать ручную разметку, иначе можно потерять важные данные.

Предположим, что у нас есть два Windows-раздела: диск C: и диск D:. Поскольку размер диска D: больше, мы будем уменьшать именно его. Выделите этот раздел и нажмите кнопку Изменить размер (рис. 1.10).

В появившемся окне с помощью ползунка установите новый размер диска D: (рис. 1.11). До этого диск D: занимал 6 Гбайт, после изменения размера — 1 Гбайт. Следовательно, мы освободили 5 Гбайт, чего вполне достаточно для установки Linux. Хотя, если дисковое пространство позволяет, я бы отвел под Linux 10 Гбайт — про запас.

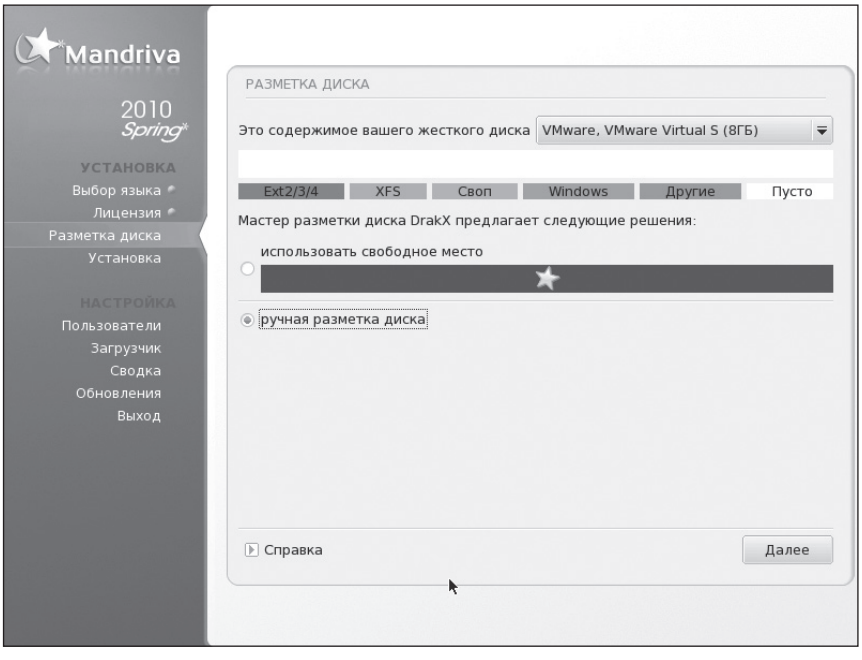

Рис. 1.9. Как будем разбивать диск?

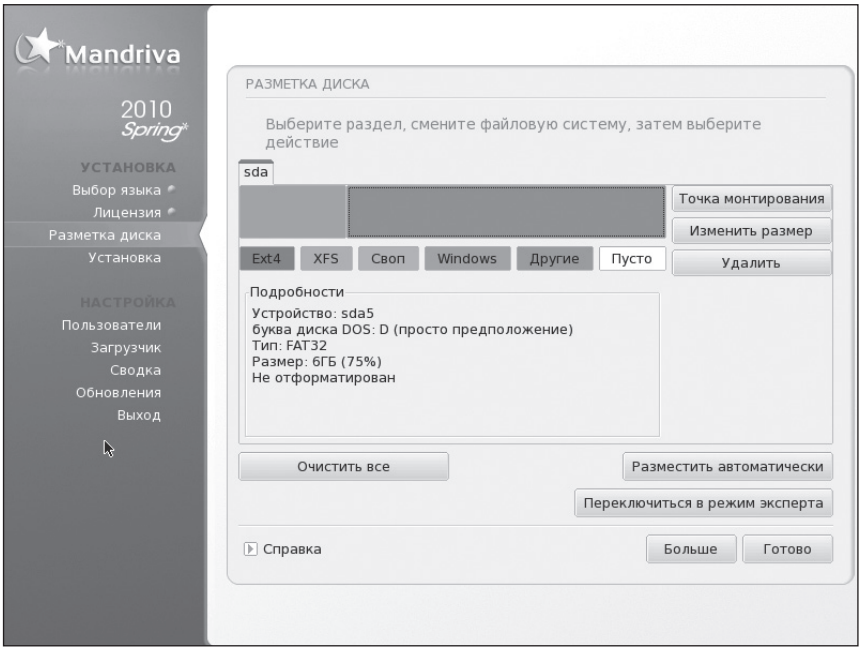

Рис. 1.10. Существующая таблица разделов

Рис. 1.11. Изменение размера Windows-раздела

Затем вы увидите, что после диска D: появилась неразмеченная область. Выделите ее и нажмите кнопку Создать (рис. 1.12).

В окне Создать новый раздел нужно установить параметры раздела. Сначала мы создадим раздел подкачки. Его размер будет зависеть от объема оперативной памяти. Если у вас мало ОЗУ, то есть от 256 до 512 Мбайт, то нужно установить размер раздела подкачки в пределах 512–1024 Мбайт. Если же у вас больше 512 Мбайт оперативной памяти, можно установить размер раздела подкачки менее 512 Мбайт.

Установить размер раздела можно с помощью ползунка. После нужно выбрать тип файловой системы Linux swap.

ПРИМЕЧАНИЕ

Более точно установить размер раздела можно с помощью кнопок ← и →. Сначала нужно выделить ползунок, а затем стрелками установить требуемый размер.

Вы увидите таблицу разделов (рис. 1.13). Опять выделите неразмеченную область и нажмите кнопку Создать.

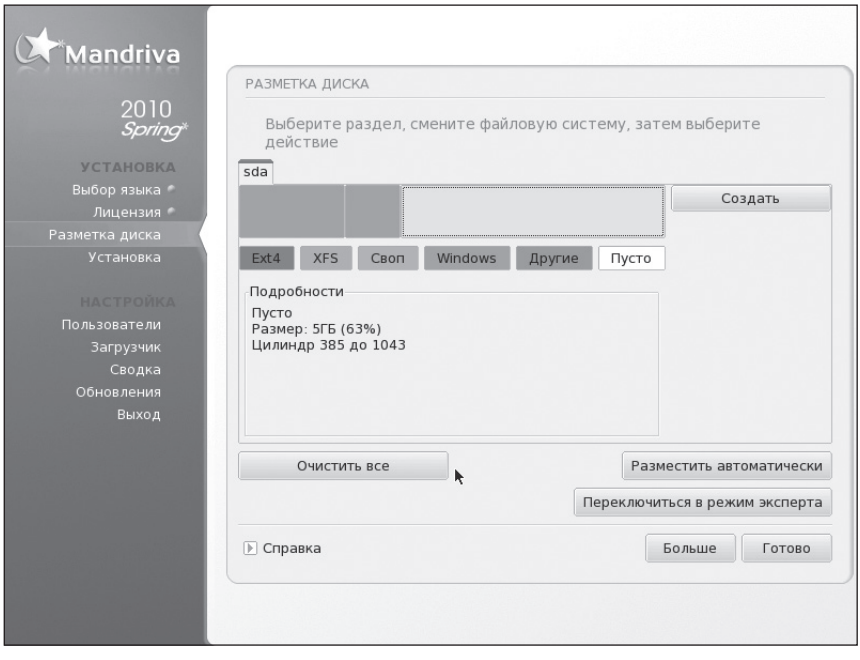

Рис. 1.12. Освободившаяся область

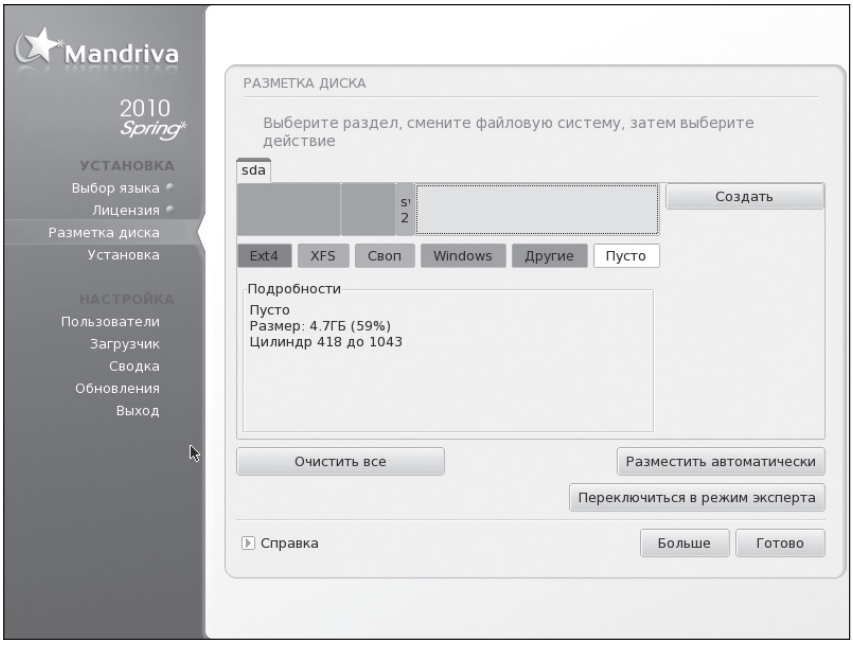

Рис. 1.13. Создан раздел подкачки

В окне Создать новый раздел установите следующие параметры (рис. 1.14):

- Размер — максимально возможный.
- Тип файловой системы — Journalised FS: ext4 (нет смысла использовать ext3 или Linux native)[1].
- Точка монтирования — /.

Окончательная таблица разделов изображена на рис. 1.15. Если все нормально, нажмите кнопку Готово для продолжения установки. В появившемся окне с предупреждением о записи таблицы разделов на жесткий диск нажмите кнопку OK.

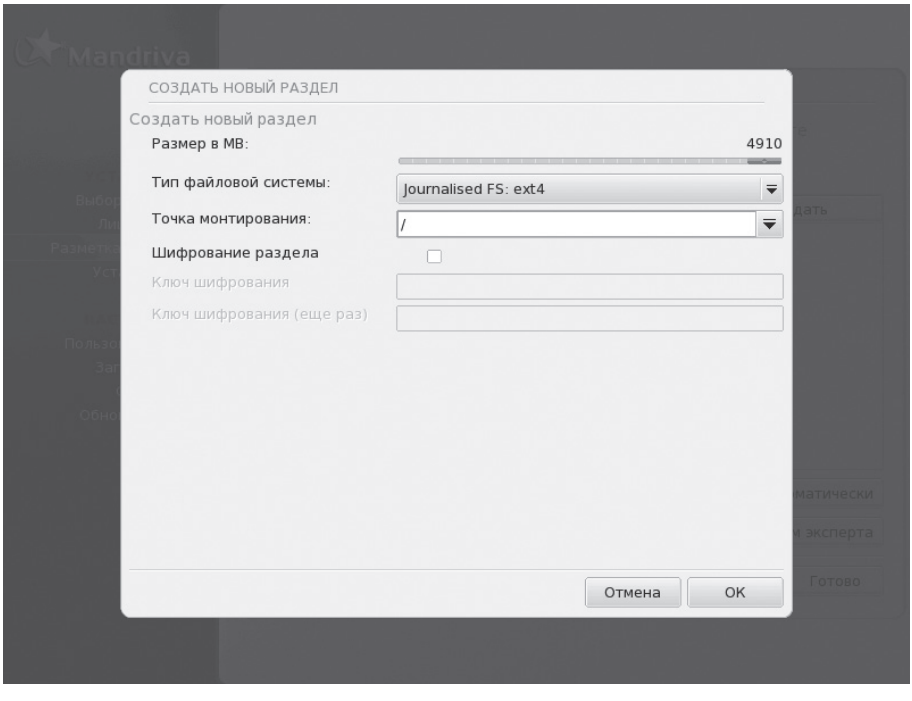

Рис. 1.14. Создание основного Linux-раздела

1.3.6. Дополнительный носитель

Если вы не скачивали Mandriva в Интернете, а покупали в интернет-магазине, вполне возможно, что за некую доплату вам продали два диска: один — установочный, второй — с дополнительными программами. При установке имеется возможность указать такой носитель (рис. 1.16). При наличии только одного диска выберите Отсутствует.

[1] По крайней мере, на этом этапе, пока не выясните, насколько соответствуют вашим задачам особенности той или иной файловой системы.

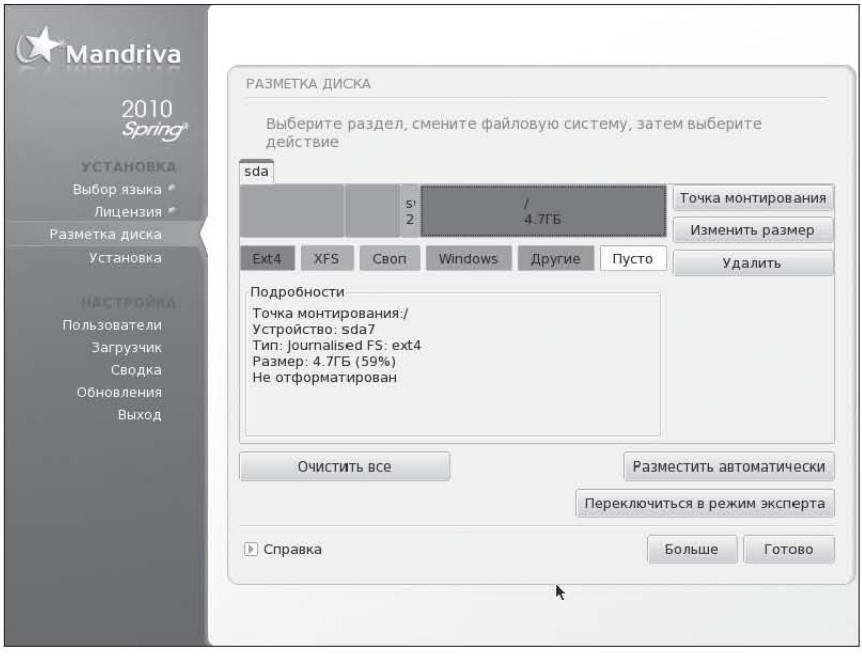

Рис. 1.15. Окончательная таблица разделов

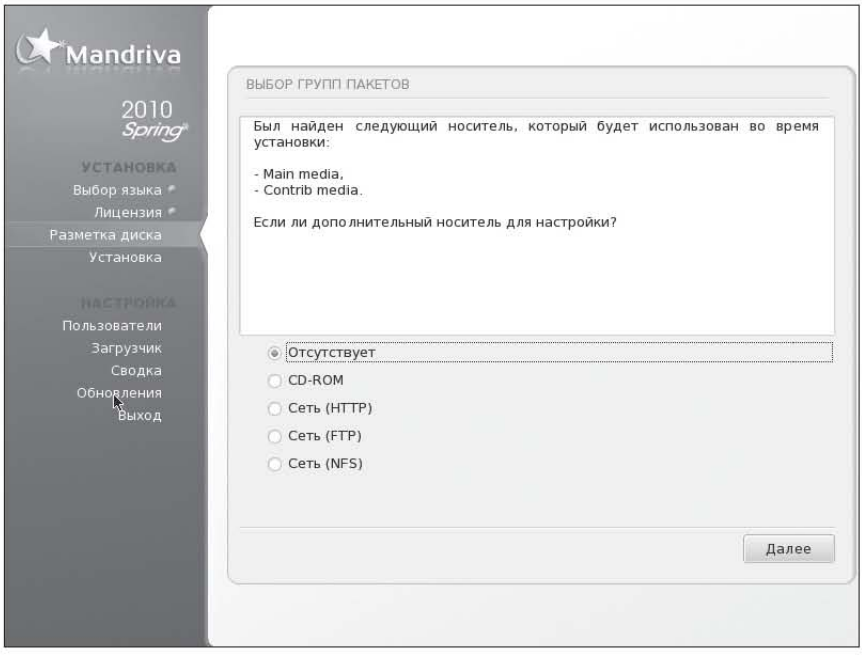

Рис. 1.16. Дополнительный носитель?

1.3.7. Выбор групп пакетов

Инсталлятор попросит вас выбрать один из профилей установки (рис. 1.17):

- KDE — установить графическую среду KDE и все приложения, использующие библиотеку Qt (на ней и основана KDE);
- GNOME — установить графическую среду GNOME и все приложения, использующие библиотеку Gtk (на ней написана GNOME);
- Выборочно — выборочная установка.

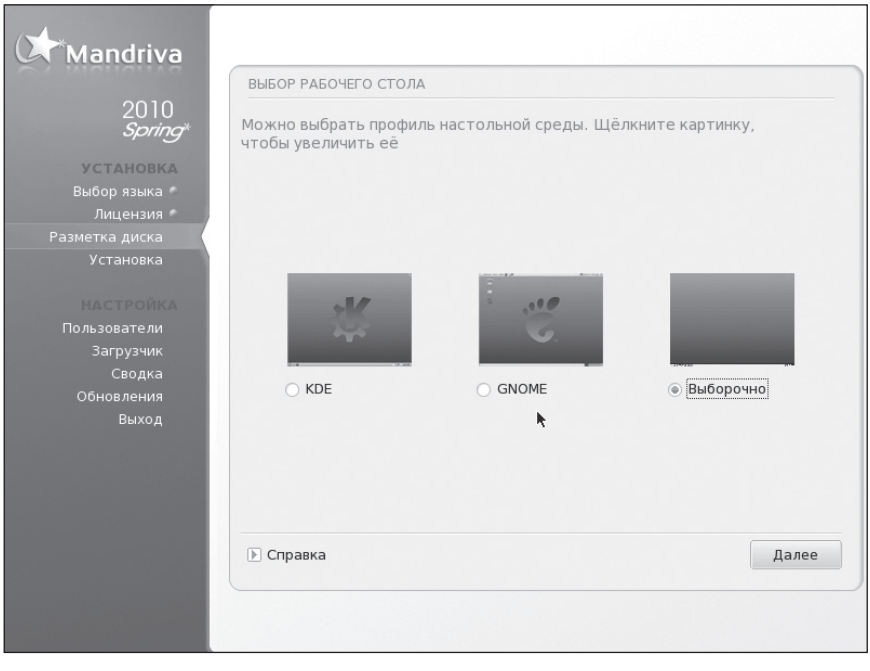

Рис. 1.17. Профиль установки

Я рекомендую выбрать третий профиль. Во-первых, выбирая один из первых двух вариантов, вы не знаете, сколько места на диске необходимо для установки. А вдруг места окажется мало? Что тогда? Я вам расскажу, что будет тогда: вам сообщат, что нет достаточно места на диске и что вам нужно начать установку заново. А этого нам не хотелось бы. Во-вторых, в случае с выборочной установкой вы можете более точно указать, какое программное обеспечение вам нужно, а какое — нет. Может, вам хочется установить обе графические среды — GNOME и KDE? Или установить утилиты разработчика (рис. 1.18)?

ПРИМЕЧАНИЕ

Для слабых компьютеров лучше всего выбрать графическую среду LXDE — она потребляет меньше системных ресурсов, чем KDE и GNOME. GNOME «легче», чем KDE, KDE — тяжеловес среди графических интерфейсов Linux. Конфигурация с LXDE займет всего 2,6 Гбайт.

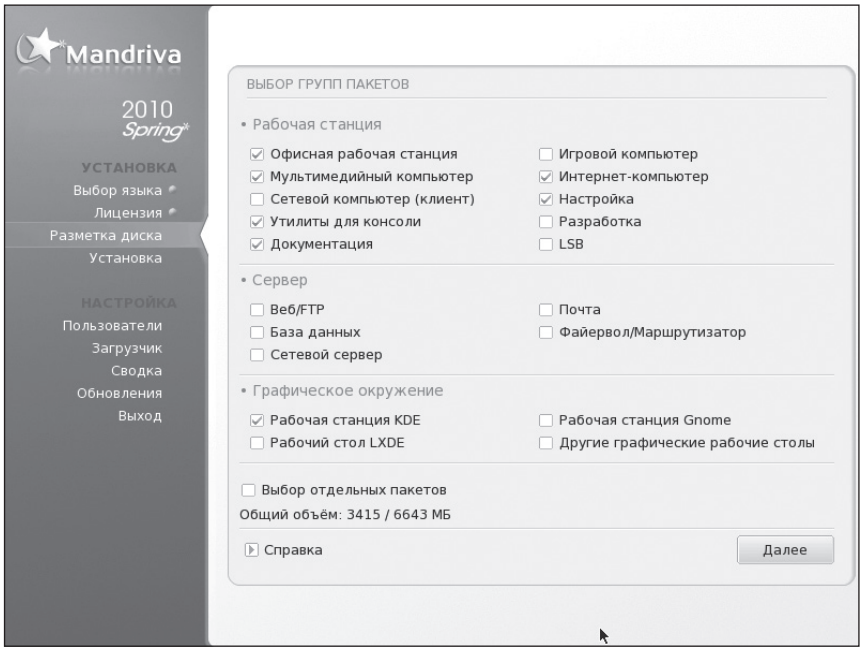

Рис. 1.18. Выбранная конфигурация занимает 3,4 Гбайт на диске

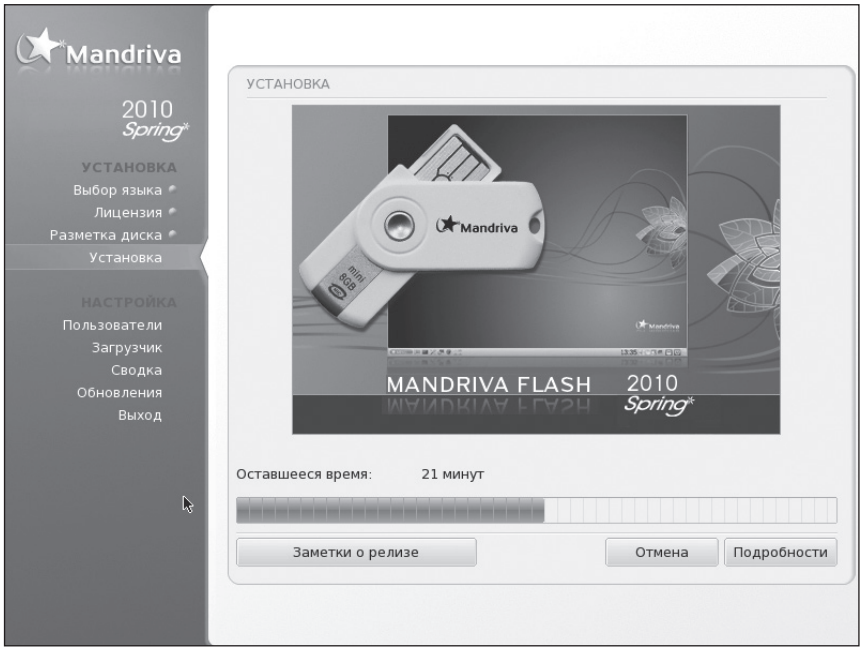

Рис. 1.19. Установка системы

Выбрав необходимые группы пакетов, нажмите кнопку Далее. Теперь вам остается только ждать, пока система установит все выбранные пакеты (рис. 1.19).

1.3.8. Пароль администратора и добавление обычных пользователей

После установки пакетов инсталлятор предложит вам указать пароль пользователя root (пользователь с максимальными правами) и добавить в систему обычного пользователя (рис. 1.20). Постарайтесь не забыть пароль root — это пароль самого главного пользователя в системе, обладающего максимальными полномочиями, именно поэтому в системе не рекомендуется постоянно работать под учетной записью root (дабы не навредить), а создать обычного пользователя для повседневной работы.

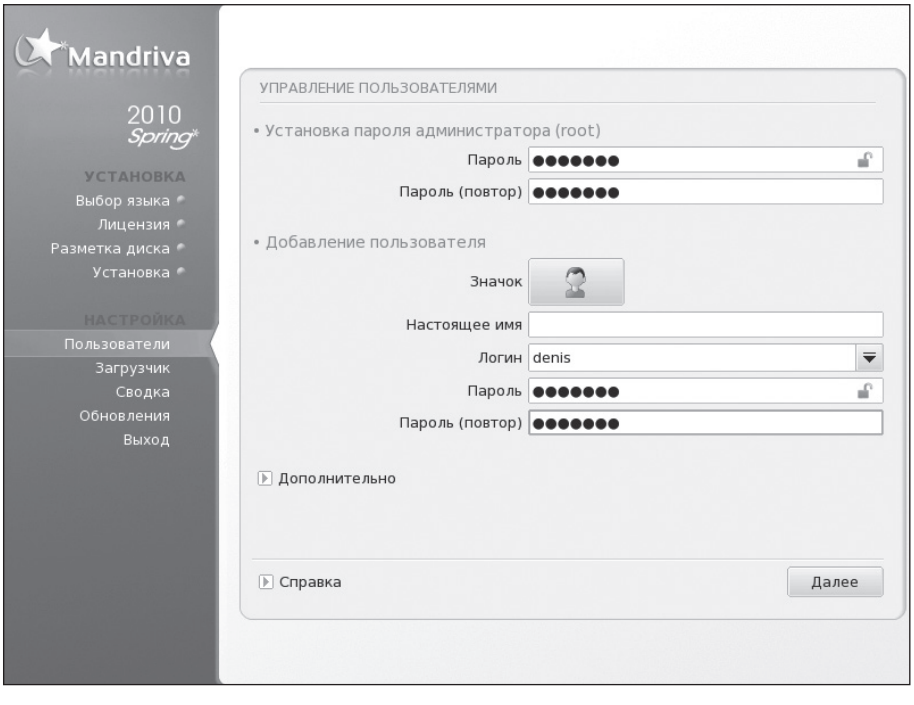

Рис. 1.20. Управление пользователями

ПРИМЕЧАНИЕ

Для управления пользователями и группами пользователей в Mandriva используется конфигуратор drakuser.

После настройки пользователей система начнет устанавливать начальный загрузчик. Процесс не требует от вас вмешательства, нужно только подождать.

1.3.9. Разрешение монитора

Выберите разрешение, поддерживаемое вашим монитором (рис. 1.23). Для комфортной работы в Linux необходимо разрешение 1024×768 или больше (например, 1280×1024), а чтобы не уставали глаза, нужно выбрать частоту обновления экрана не менее 70 Гц (правда, частота обновления имеет значение только в случае со старыми ЭЛТ-мониторами).

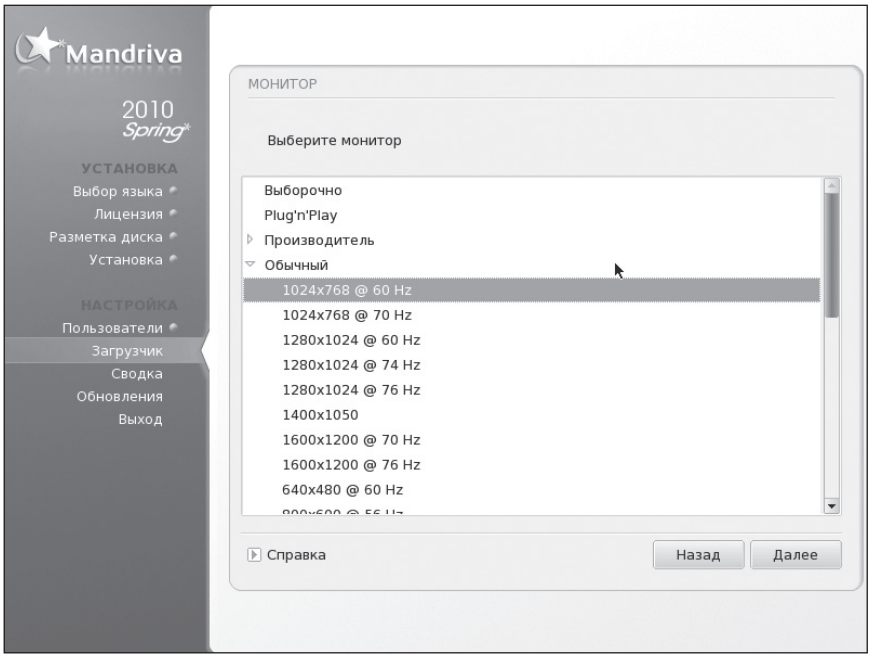

Рис. 1.21. Выбор разрешения монитора

1.3.10. Сводка по системе

Инсталлятор отобразит сводку по системе (рис. 1.22), а именно перечень параметров и краткое описание значения каждого параметра. Обычно сейчас изменять эти параметры не нужно, можно разве что выбрать часовой пояс и установить точное местное время.

Для продолжения нажмите Далее. Инсталлятор спросит вас, хотите ли вы установить обновления пакетов. Нужно отметить Нет хотя бы потому, что мы еще не настраивали подключение к Интернету (рис. 1.23).

Программа установки поздравит вас с установкой Linux Mandriva на ваш компьютер, но не спешите нажимать кнопку Перезагрузка (рис. 1.24). Перед ее нажатием нужно извлечь из привода инсталляционный диск, иначе установка Mandriva будет запущена еще раз!

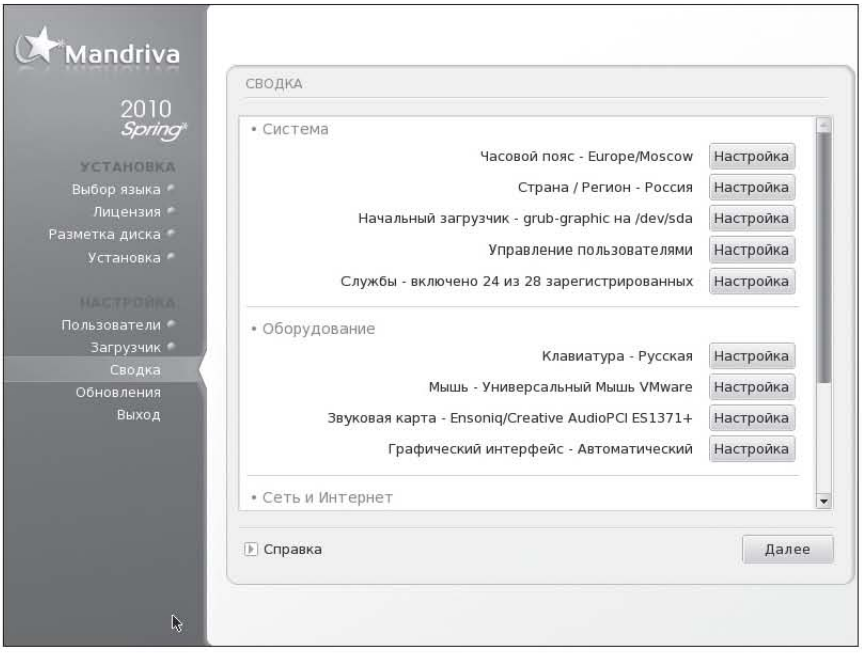

Рис. 1.22. Сводка по системе

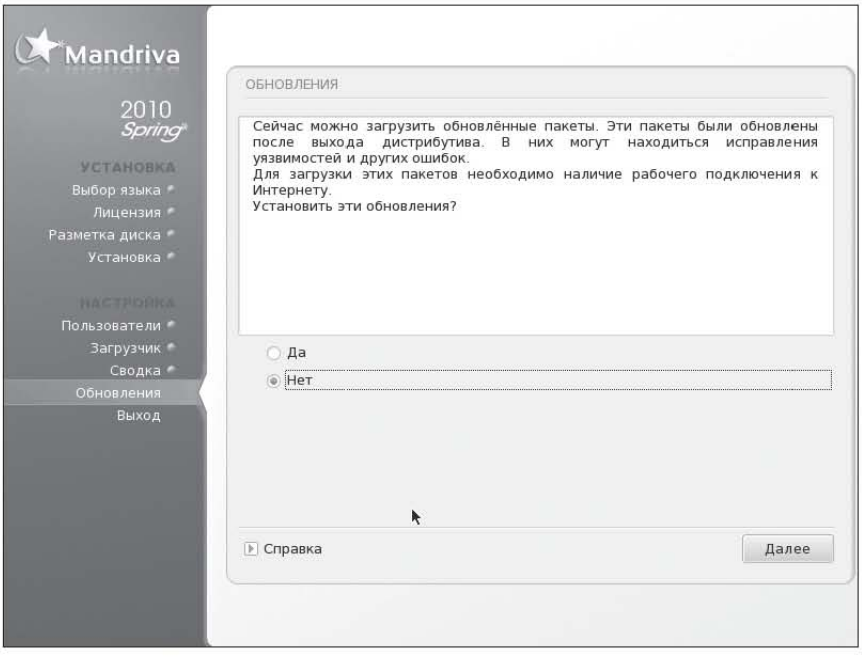

Рис. 1.23. Отказываемся от установки обновлений

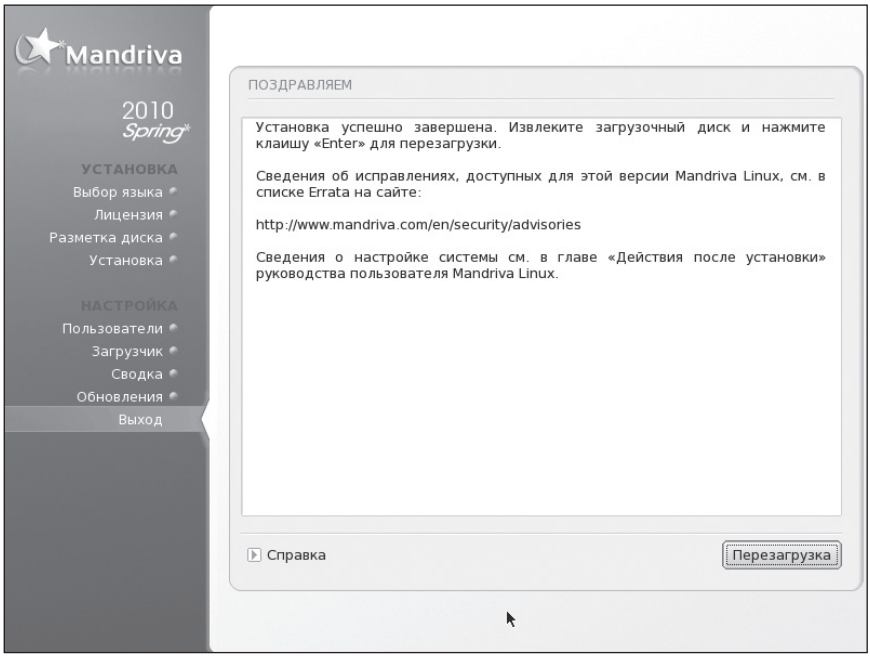

Рис. 1.24. Завершение установки

Рис. 1.25. Система успешно установлена

1.4. После перезагрузки...

После перезагрузки вы увидите сообщение о том, что Mandriva была успешно установлена на ваш компьютер (рис. 1.25). Также вам будет предложено пройти бесплатную регистрацию, но в большинстве случаев вы не сможете этого сделать, поскольку Интернет еще не настроен, так что от регистрации придется отказаться.

После этого вы увидите окно входа в систему. Вам нужно ввести имя пользователя и пароль (рис. 1.26), указанный при установке системы. Вход в систему как пользователя root в графическом режиме запрещен (позже мы поговорим, как обойти это ограничение).

Рис. 1.26. Вход в систему

Список выбора сеанса (на рис. 1.26 в нем выбран сеанс GNOME) позволяет выбрать графическую среду — KDE или GNOME (если установлены обе среды), а кнопка Параметры выключения (в нижнем правом углу) отображает меню вариантов завершения работы системы (рис. 1.27)

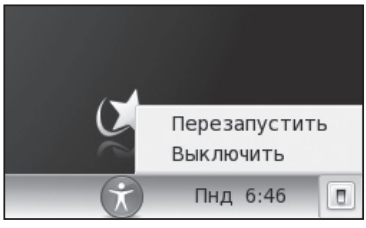

Рис. 1.27. Меню завершения работы

На этом установка системы завершена. Установка других дистрибутивов, как уже было отмечено, подобна установке Mandriva, но везде есть свои отличия. Довольно похожей (по этапам установки) выглядит установка openSUSE. Установка Fedora отличается тем, что настройка системы выполняется после первой перезагрузки: сначала система устанавливается на жесткий диск, а дополнительная настройка (вроде добавления пользователей) осуществляется после перезагрузки. С установкой Ubuntu справится даже школьник — она состоит всего из семи шагов, самый сложный среди которых — это разметка диска.

Установка FreeBSD 2

2.1. Перед установкой

Перед установкой FreeBSD нам нужно найти компьютер, на который можно было бы установить FreeBSD (читайте — ознакомиться с системными требованиями), сделать резервную копию данных этого компьютера и, конечно же, загрузить саму FreeBSD.

Начнем с системных требований. Для установки FreeBSD подходит абсолютно любой компьютер (в пределах разумного), понятно, что на компьютер с процессором 80386 можно даже не пытаться установить FreeBSD, хотя не уверен, что вы вообще найдете такой компьютер. А если серьезно, то FreeBSD поддерживает архитектуры Intel x86, AMD64, Intel EM64T, IA-64, ARM, PowerPC, sun4v, SPARC64. Не будет проблем и с поддержкой многопроцессорных SMP-систем. Другими словами, FreeBSD можно установить как на старенький Pentium, который и использовать сейчас нельзя, и выбросить жалко, так и на современный и модный SPARC64.

Что же касается памяти, то для минимальной установки должно хватит 64–128 Мбайт. Это если у вас совсем старый компьютер. А на более или менее современном компьютере сейчас установлено как минимум 256 Мбайт, что более чем достаточно для запуска FreeBSD.

FreeBSD поддерживает машины с объемом оперативной памяти более 4 Гбайт. Но помните, что кроме самой операционной системы на лимит памяти влияет используемая архитектура. Например, для 32-разрядной i386 верхний предел — 4 Гбайт. Если вы умудрились на столь несовершенный компьютер установить более 4 Гбайт, тогда вам нужно включить PAE (Physical Address Extension) — пересобрать ядро с опцией PAE. А вот с архитектурой AMD64 никаких проблем не будет — она изначально поддерживает до 1 Тбайт (да, это не опечатка) оперативной памяти. Так что при выборе сервера нужно учитывать и архитектуру компьютера. Иногда получается так — взяли средний компьютер, установили на него FreeBSD, а затем захотели развернуть мощный сервер баз данных Oracle. Не хватило «оперативки», попытались увеличить, а система «не видит» добавленные модули.

Пространство на жестком диске тоже зависит от задач сервера. Если вы собираетесь строить небольшой интернет-шлюз, то хватит и 10 Гбайт — это даже с запасом. Для FTP-сервера, сервера баз данных и тем более для сервера хостинг-провайдера нужно гораздо больше. Сколько именно? Давайте посчитаем. Нужно создать сервер для хостинг-провайдера, где другие пользователи будут размещать собственные сайты (а очень часто сайт — это не только файлы, но и база данных). Пусть у нас есть жесткий диск на 160 Гбайт. 10 Гбайт резервируем для самой системы. Остается 150 Гбайт. Средний тарифный план на сегодняшний день предлагает 500 Мбайт для хостинга. Выходит, что на таком сервере смогут разместить свои сайты 300 пользователей. Много это или мало, решайте сами. Как по мне, то очень мало. Для такой задачи нужен жесткий диск объемом 1–2 Тбайт.

По следующей ссылке вы можете ознакомиться со списком совместимого «железа» для текущей версии FreeBSD:

http://www.freebsd.org/doc/en_US.ISO8859-1/books/faq/hardware.html

Итак, компьютер мы уже выбрали. Если на его жестком диске есть какие-нибудь данные, то на всякий случай сделайте резервную копию (сегодня они вам не нужны, а завтра окажется, что вчера вы удалили что-то важное) — мы будем устанавливать FreeBSD на весь диск, поэтому все данные после установки будут утеряны.

Осталось скачать саму FreeBSD. Какую версию выбрать? На сегодняшний день сами разработчики FreeBSD рекомендуют выбирать или версию 8.1 или версию 7.3 — это наиболее надежно работающие версии. Версия 9.0 на момент написания этих строк находится на стадии CURRENT и поэтому ее нельзя назвать стабильной. В ветку CURRENT помещаются все запланированные нововведения. Потом начинается тестирование. Понятно, что не все новые возможности стабильно работают. Все, что работает стабильно, помещается в ветку STABLE. Затем STABLE тестируют release-инженеры и независимые пользователи, все найденные ошибки удаляются, и выпускается релиз — RELEASE. Я вам вкратце пояснил модель разработки FreeBSD, чтобы вы могли ориентироваться при выборе ветки.

А теперь мое послание в будущее: даже если вы купите эту книгу спустя полгода или год и к тому времени будет выпущен релиз 9-й версии, все описанное в этой книге будет верно и для 9-й версии. Вы даже не заметите разницы, если не считать номера версии.

Скачать ISO-образы можно с FTP-сервера ftp.freebsd.org:

ftp://ftp.freebsd.org/pub/FreeBSD/releases/

Сначала перейдите в каталог, соответствующий вашей архитектуре, например `amd64`, затем в каталог `ISO-IMAGES`. Далее нужно перейти в каталог, соответствующий желаемой версии — `8.1`. Осталось только скачать нужный образ:

```
FreeBSD-8.1-RELEASE-amd64-disc1.iso
FreeBSD-8.1-RELEASE-amd64-dvd1.iso.gz
```

Первый занимает 642 Мбайт, второй — 1,9 Гбайт. Понятно, что лучше скачать второй образ. Но если вы планируете установить систему на старый компьютер, где установлен привод CD-ROM, а не DVD-ROM, выберите первый образ — вы сможете записать его на диск CD-R. А вот второй образ можно записать только на DVD-R. А если у вашего компьютера DVD-привод, экономить не нужно — все равно потом все необходимое программное обеспечение придется загружать из Интернета.

Второй образ запакован архиватором gz, поэтому его нужно распаковать. Этот архив можно распаковать любым архиватором, поддерживающим формат ZIP. Я распаковал встроенным архиватором Total Commander (после распаковки он займет 2,1 Гбайт). Думаю, у вас проблем с этим не возникнет. Далее счастливые обладатели Windows 7 или собранного мной Linux-дистрибутива Denix (http://denix.dkws.org.ua) могут щелкнуть правой кнопкой мыши на распакованном образе и выбрать команду записи на диск (рис. 2.1).

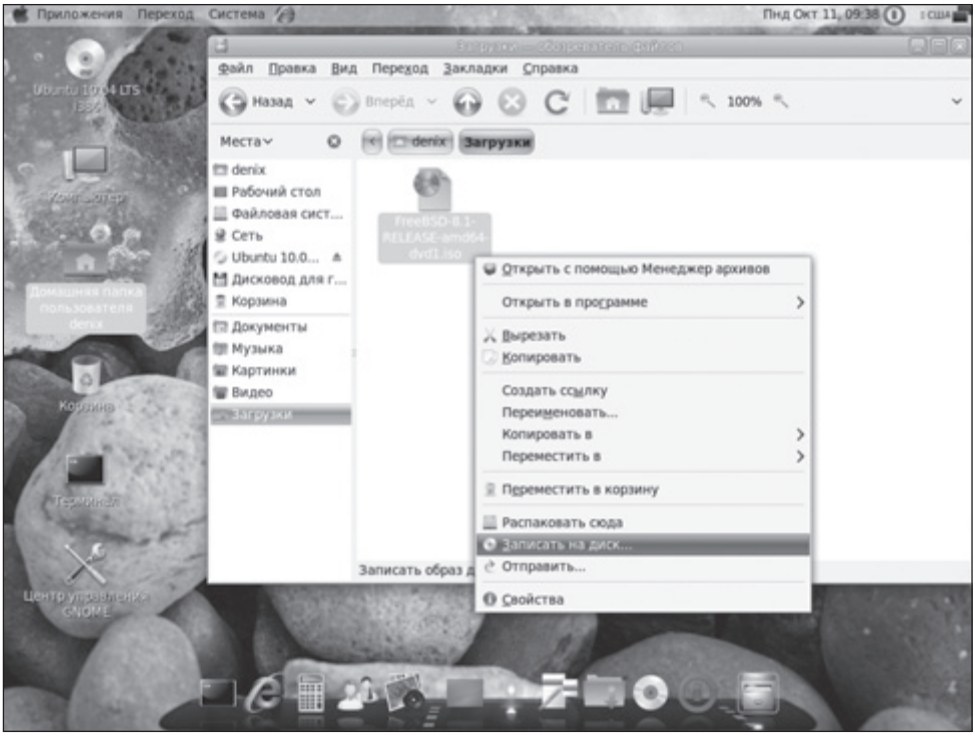

Рис. 2.1. Запись ISO-образа на болванку в Denix

Перезагрузите компьютер, зайдите в BIOS SETUP, измените порядок загрузки устройств (нужная опция может называться Boot Order, Boot Sequence, First Boot Device и т. п.), выберите CD-ROM в качестве первого загрузочного устройства. Вставьте только что записанный диск в привод компьютера, нажмите F10 для сохранения параметров BIOS.

2.2. Установка системы

2.2.1. Загрузка с диска. Выбор типа установки

После загрузки с диска вы увидите меню загрузчика (рис. 2.2). В большинстве случаев нужно выбрать пункт 1 или просто нажать Enter. Если у вас компьютер с процессором AMD64 и видеокартой от NVIDIA, тогда выберите пункт 2 — нужно отключить ACPI.

Рис. 2.2. Меню загрузчика FreeBSD

Начнется загрузка FreeBSD (рис. 2.3), придется немного подождать. Первым делом программа установки попросит вас выбрать страну и раскладку клавиатуры. Думаю, с этим вы справитесь сами. Даже если что-то выберете не так, особой роли не играет, потому что мы будем заниматься русификацией системы самостоятельно, а толку от выбранной раскладки во время установки мало.

После выбора раскладки вы увидите главное меню программы установки (рис. 2.4)

Меню программы установки позволяет выбрать тип установки (Standard, Express, Custom), а также установить некоторые параметры установки (команда Configure). Также можно использовать установочный диск для восстановления системы (команда Fixit), но в данный момент нас больше интересует установка системы.

Программа установки позволяет выбрать один из трех типов установки: стандартный (Standard), экспресс (Express) и пользовательский (Custom). Первый тип установки больше всего подходит для начинающих пользователей. Перед каждым этапом установки появляется информационное окошко, где вкратце пояснено, что нужно сделать. Перед каждым существенным (читайте — потенциально опасным) действием выводится предупреждение. А после самой установки системы запускается процесс ее настройки. Вам будет предложено настроить много чего — от настройки сети до установки пакетов. Могу поспорить, что под конец установки вам надоест отвечать на вопросы нажатием Yes или No и вы пожалеете,

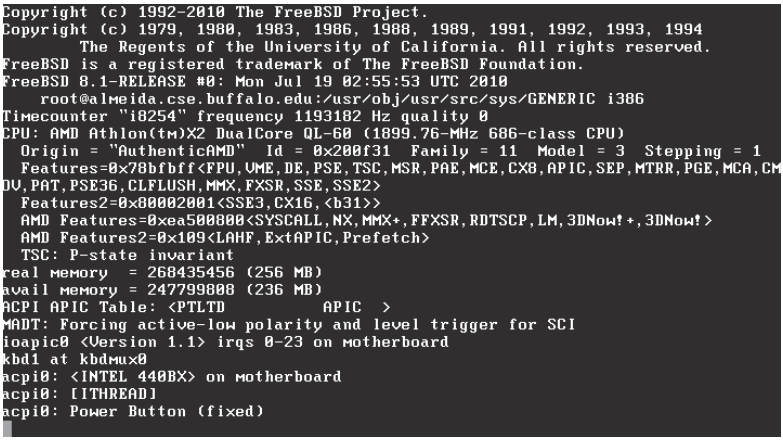

Рис. 2.3. Загрузка системы

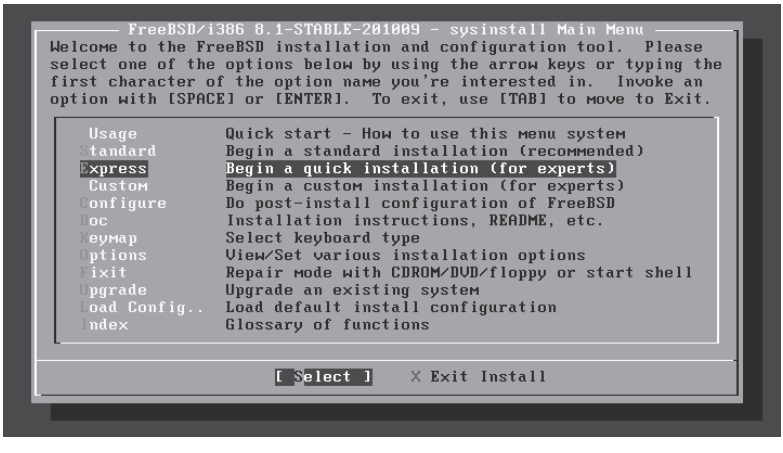

Рис. 2.4. Главное меню программы установки sysinstall

что выбрали этот тип установки. Третий тип установки тоже не очень удобный. Зато мне больше всего нравится экспресс-установка. Да, первый вариант как бы предпочтительнее из-за навязчивых рекомендаций и настройки после установки (постинсталляционной настройки), но ведь у вас есть эта книга. Я вам и так расскажу, что нужно сделать на том или ином этапе установки, а всей следующей после установки системы настройке посвящена большая часть этой книги — мы только и будем делать, что настраивать систему. Не играть же в тетрис?

Минимальную постинсталляционную настройку после экспресс-установки мы выполним в этой главе, и, поверьте, это будет намного быстрее и полезнее. Мы настроим нашу сеть на использование DHCP, установим пароль root и добавим обычного пользователя. Это все самое полезное, что предлагает инсталлятор после установки системы, все остальное — пустая трата времени, потому что нельзя нормально настроить тот же FTP-сервер средствами

конфигуратора, а редактированием его конфигурационного файла можно добиться лучшего результата.

2.2.2. Использование fdisk

Сразу после выбора типа установки (при условии, что выбрали экспресс-установку) запустится редактор разделов жесткого диска — `fdisk` (рис. 2.5). А теперь нужно сделать небольшое отступление и поговорить о разделах. Мы знаем, что один жесткий диск можно разбить на несколько логических дисков или разделов (partitions). Во FreeBSD используется другой термин для логических дисков — слайс (slice). Внутри слайса можно создать несколько разделов. Это не разделы жесткого диска, это внутреннее представление BSD-слайса. Можно сказать, что BSD-слайс похож на расширенный раздел DOS — ведь внутри расширенного раздела можно создавать обычные разделы. Точно такая же схема используется и во FreeBSD.

Жесткие диски во FreeBSD получают имена:

- `adN` — IDE-диск, N — номер диска
- `daN` — SCSI-диск, N — номер диска

Имя слайса (раздела в терминологии Windows и Linux) назначается так: `<имя_диска>sM`, M — номер слайса (нумерация начинается с 1). Например, имя `ad0s1` принадлежит первому слайсу на IDE-диске, подключенном как Primary Master. В случае с IDE-диском номер N зависит от подключения диска:

- `ad0` — Primary Master;
- `ad1` — Primary Slave;
- `ad2` — Secondary Master;
- `ad3` — Secondary Slave.

Разделам внутри слайса присваиваются имена `a`, `b`, `c` и т. д. Вот пример имени первого раздела на первом слайсе на первом IDE-диске: `ad0s1a`. Понимаю, что вам сложно привыкнуть, но, думаю, скоро разберетесь, тем более что в этой книге мы подробно рассмотрим устройство файловой системы BSD. А пока, чтобы не было путаницы, давайте разделы внутри слайсов будем называть BSD-разделами, чтобы сразу было ясно, о каких разделах идет речь.

Итак, вернемся к разметке диска. Поскольку мы решили использовать для FreeBSD весь жесткий диск, то можете смело нажать `A` для автоматической разметки. А как иначе? Ведь мы создаем сервер, а сервер должен быть включенным постоянно. На нем просто не может быть установлена еще одна операционная система, иначе это будет уже не сервер, а экспериментальная площадка. Если вам нужна такая площадка, обратитесь к приложениям, в которых мы рассмотрим совместное сосуществование FreeBSD с Windows и Linux.

При выборе автоматической разметки все слайсы (Windows-слайсы, Linux-слайсы — привыкайте к новой терминологии) будут удалены, зато будет создан один слайс на весь жесткий диск. На рис. 2.6 изображен результат автоматической разметки.

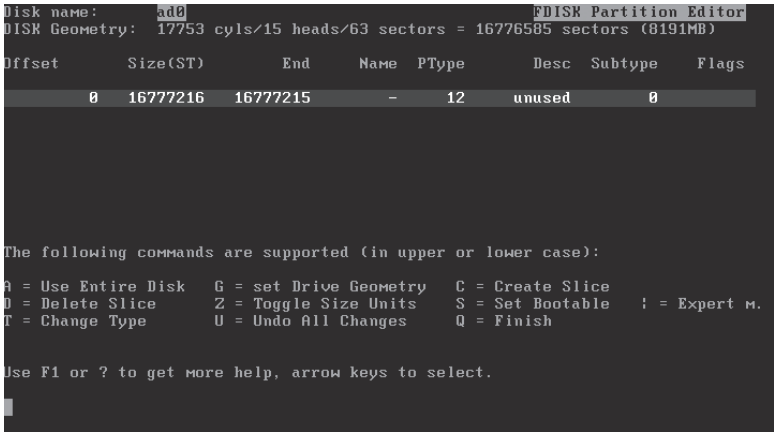

Рис. 2.5. Программа fdisk

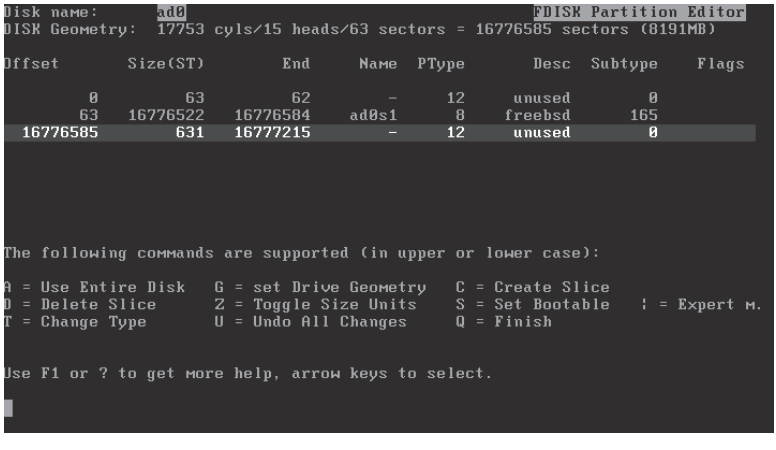

Рис. 2.6. Результат автоматической разметки жесткого диска

Нажмите Q для сохранения изменений и перехода к следующему этапу.

2.2.3. Выбор загрузчика

После разметки диска нужно будет выбрать загрузчик операционной системы (рис. 2.7), вам будет предложено три варианта: Standard, BootMgr и None. Последний вариант — это вообще отсутствие какого-либо загрузчика. После установки вы не сможете запустить только что установленную систему. Может, вы хотите настроить загрузку FreeBSD через другой загрузчик. Тогда, безусловно, нужно выбрать None, но вы должны знать, что делаете.

Первый вариант — это самый простой загрузчик: он находит ядро, загружает его в память и передает ему управление. Скучно и не интересно. При загрузке с диска вы видели загрузчик с меню. Это и есть BootMgr. Довольно удобный и функциональный загрузчик, поэтому его и выбираем.

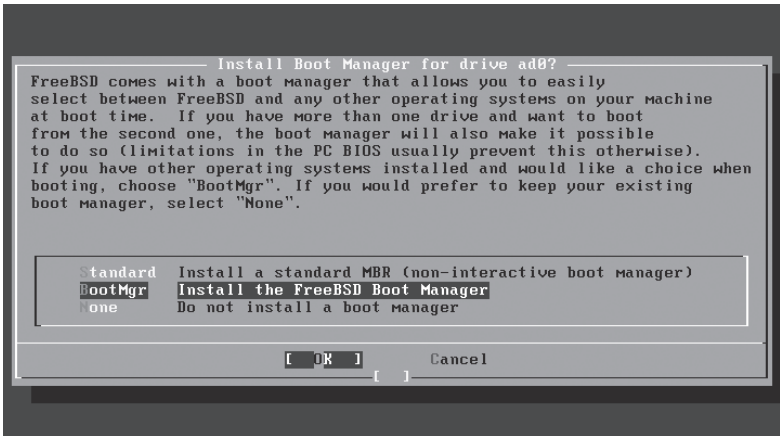

Рис. 2.7. Выбор загрузчика

2.2.4. Разметка BSD-слайса

Следующий этап — это разметка BSD-слайса, то есть создание BSD-разделов внутри слайса (рис. 2.8). Как по мне, можно смело нажать A для автоматической разметки — тут вы уже ничего не сломаете, пусть за вас всю работу сделает программа установки. Стоит отметить, что мы устанавливает FreeBSD на систему с жестким диском размером всего 8 Гбайт — при разметке диска нужно учитывать его размер.

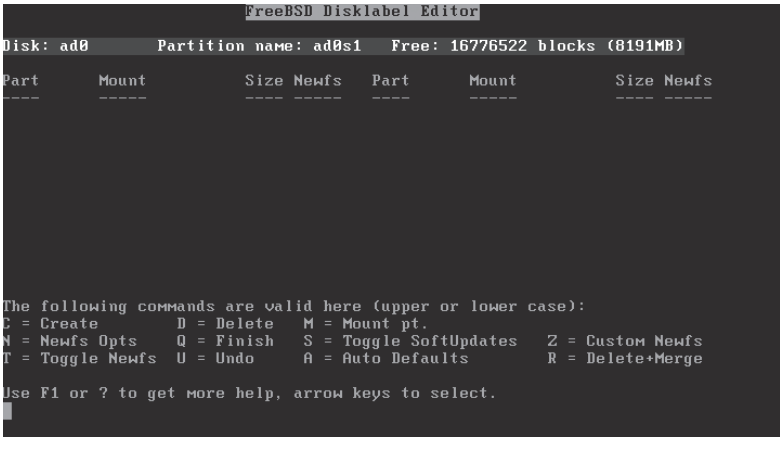

Рис. 2.8. Пока BSD-разделы не созданы

На рис. 2.9 приведен результат автоматической разметки слайса. Программа Disklabel Editor создала пять BSD-разделов. Каждый из разделов монтируется к своей точке монтирования. Точка монтирования — это каталог корневой файловой системы, через который можно обратиться к данным, записанным на разделе.

Представим, что вы записали файл на BSD-раздел ad0s1f. После того как вы подмонтируете этот BSD-раздел к каталогу /usr, через каталог /usr можно будет обратиться к записанному файлу. Конечно, объяснение довольно грубое, но когда мы будем изучать файловую систему UNIX, мы еще поговорим о монтировании.

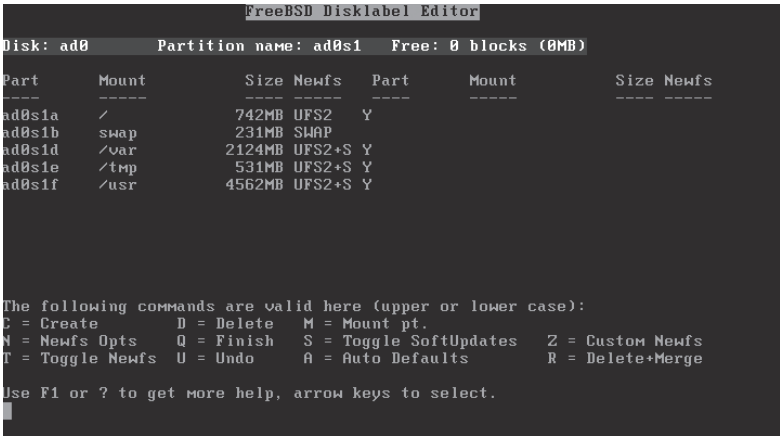

Рис. 2.9. Результат автоматической разметки слайса

Для самого первого раздела ad0s1a было отведено 742 Мбайт. В этом разделе будут основные утилиты и основные конфигурационные файлы FreeBSD. Больше места для этого раздела и не нужно. Первый раздел монтируется к корневому каталогу (/).

Второй раздел используется как раздел подкачки (swap). Размер раздела — 231 Мбайт. Плюс к этому 256 Мбайт физической памяти (на машине, на которую производилась установка системы, было установлено именно 256 Мбайт ОЗУ), выходит 487 Мбайт виртуальной памяти.

Для раздела d (раздел c во FreeBSD вообще не используется, это имя зарезервировано) отведено 2,1 Гбайт. Если не планируете создавать сервер баз данных, то 2,1 Гбайт вполне должно хватить. Каталог /var, к которому монтируется этот раздел, обычно используется для хранения баз данных, почтовых ящиков пользователей, протоколов системы и других часто изменяющихся данных. Но не забывайте, что у нас тестовый жесткий диск всего 8 Гбайт, если было бы 80 Гбайт, то все эти числа можно было бы умножить на 10. Так что программа Disklabel Editor вполне нормально справляется с поставленной задачей.

Для каталога временных файлов /tmp (раздел ad0s1e) было «отрезано» 531 Мбайт. Будем надеяться, что нам этого хватит. А все оставшееся место было отведено под последний BSD-раздел, который будет монтироваться к каталогу /usr. В этот каталог устанавливаются программы, поэтому правильно, что было отведено столько места. Ведь только коллекция портов занимает около 450 Мбайт.

Если вы не согласны с автоматической схемой разметки слайса, нажмите U для отмены всех изменений, а затем используйте клавишу C для создания разделов вручную. При создании раздела нужно будет ввести размер раздела, используйте

модификатор M для указания того, что размер вводится в мегабайтах, например 700M. После указания размера нужно будет выбрать тип раздела — FS (файловая система) или SWAP (подкачка). Не нужно создавать несколько разделов подкачки, хватит одного, и если нет особых предпосылок, то не нужно устанавливать для него размер больше 512 Мбайт. После выбора типа раздела нужно указать точку монтирования. Всего вам нужно создать 5 разделов: один подкачки и по одному для точек монтирования /, /var, /tmp и /usr.

Когда будет все готово, нажмите Q для перехода к следующему этапу установки.

2.2.5. Выбор программ для установки

После разметки слайса нужно выбрать «дистрибутивный набор программ» (distribution set) — набор программ, который будет установлен (рис. 2.10). При условии наличия достаточного количества дискового пространства можно выбрать All для установки всех наборов. Можно выбрать Custom для самостоятельного выбора групп программного обеспечения. Остальные наборы вряд ли вам понравятся:

- Developer — содержит помимо всего прочего все исходные коды, нам это явно не нужно;
- Kern-Developer — вместо всех исходников содержит только исходные коды ядра;
- User — типичная рабочая станция;
- Minimal — минимальная установка, минимум места на диске, но почти ничего не установлено.

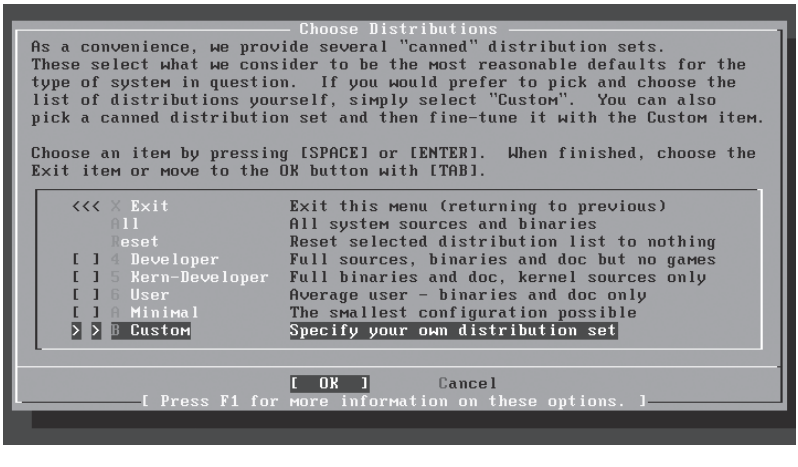

Рис. 2.10. Выбор дистрибутивного набора FreeBSD

Как видите, ни один из вариантов не подходит для создания сервера, поэтому или выбирайте All или Custom. При выборе Custom вам нужно будет выбрать, что именно вы хотите установить. Обязательно установите base, kernels, man и ports (рис. 2.11). Остальное — на свое усмотрение. Когда выберете kernels, нужно бу-

дет указать, какие ядра вы желаете установить, обязательно выберите GENERIC (рис. 2.12). Поскольку я выбрал образ для CD-ROM, то мне доступно только одно ядро, но и его нужно обязательно выбрать.

Рис. 2.11. Выбор групп дистрибутивов вручную

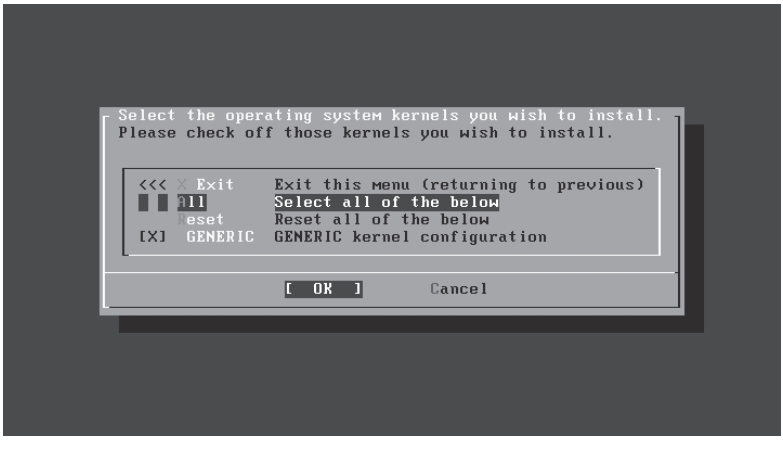

Рис. 2.12. Выбор ядер

2.2.6. Завершение установки

Вам осталось только выбрать носитель установки (рис. 2.13). Выбирайте CD/DVD, затем вы увидите устрашающее предупреждение (рис. 2.14), выберите Yes и наслаждайтесь процессом установки.

По окончании установки вы увидите окошко, изображенное на рис. 2.15. В нем нужно выбрать No, вы окажетесь в главном меню Sysinstall, нажмите кнопку Exit install (рис. 2.16). После этого система спросит вас, желаете ли вы перезагрузить

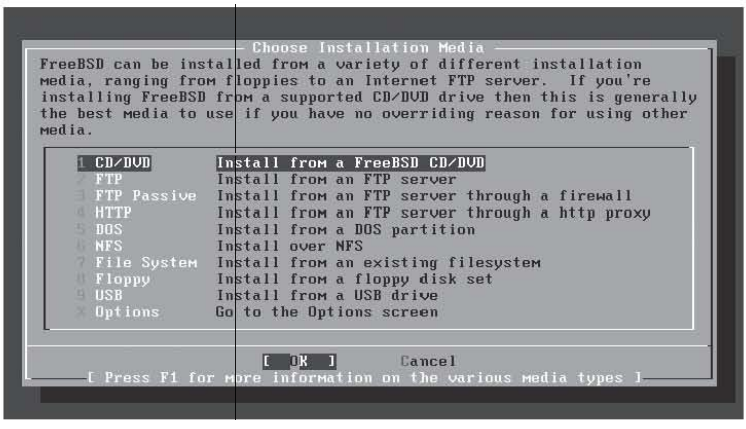

Рис. 2.13. Выбор носителя

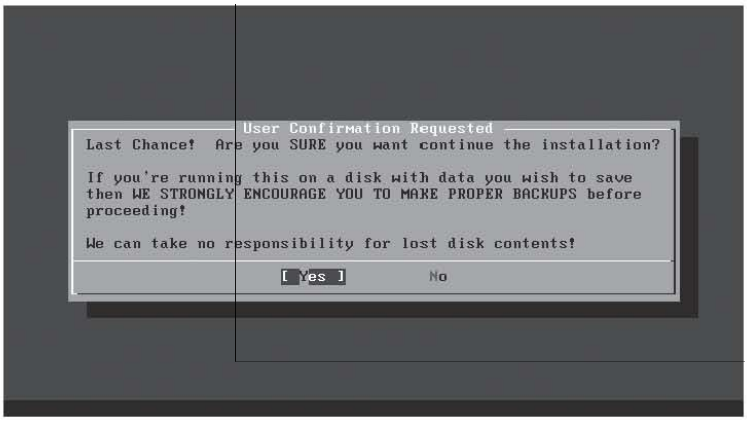

Рис. 2.14. Предупреждение о потере данных

Рис. 2.15. Просмотреть еще раз параметры установки?

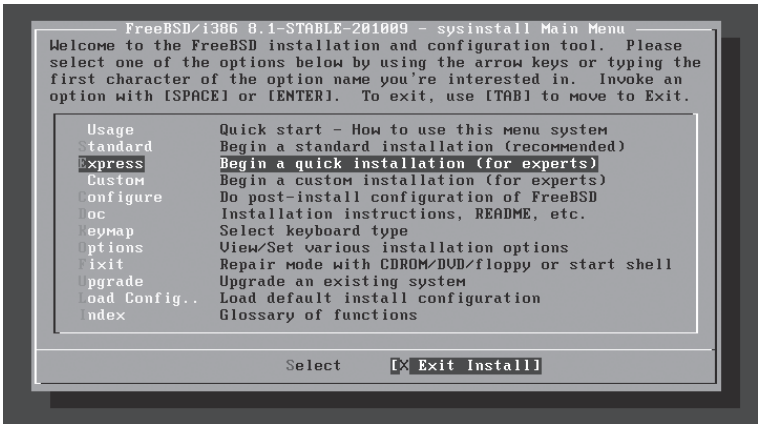

Рис. 2.16. Нажмите Exit install

компьютер, выберите Yes. Во время перезагрузки не забудьте извлечь установочный диск, чтобы система повторно не загрузилась с него.

2.3. Небольшая настройка после установки

Компьютер будет перезагружен, после перезагрузки вы увидите приглашение войти в систему (рис. 2.17).

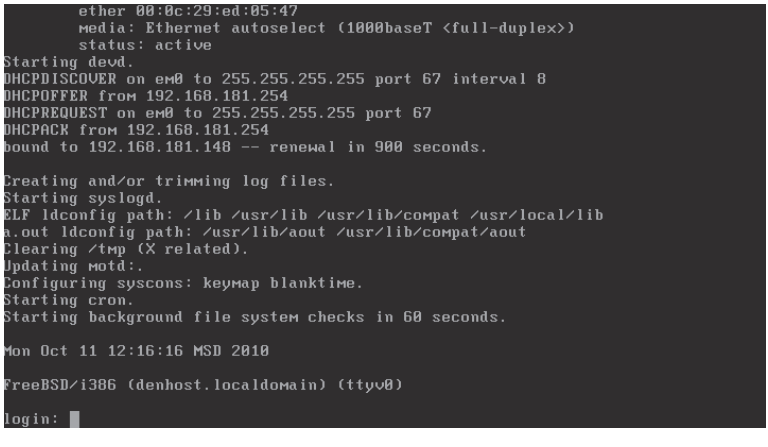

Рис. 2.17. Приглашение войти в систему

Введите имя пользователя root, в качестве пароля ничего вводить не нужно — по умолчанию у пользователя root нет пароля.

Я пообещал вам, что после установки мы:

- настроим сеть на использование DHCP-сервера;
- установим пароль пользователя root;
- добавим обычного пользователя.

Приступим к настройке. Введите команду для редактирования основного конфигурационного файла операционной системы:

```
# ee /etc/rc.conf
```

ПРИМЕЧАНИЕ

Давайте договоримся сразу: если команда должна быть выполнена от имени root (это пользователь, обладающий максимальными полномочиями в UNIX), то перед ней будем указывать **#**. Если же команду можно выполнить от имени обычного пользователя, то перед командой или вообще ничего не будем указывать или будем указывать **$**[1].

Добавьте в него две строки (рис. 2.18):

```
hostname="имя_компьютера"
ifconfig_em0="DHCP"
```

Для выхода из редактора нажмите Esc, затем a, потом еще раз a.

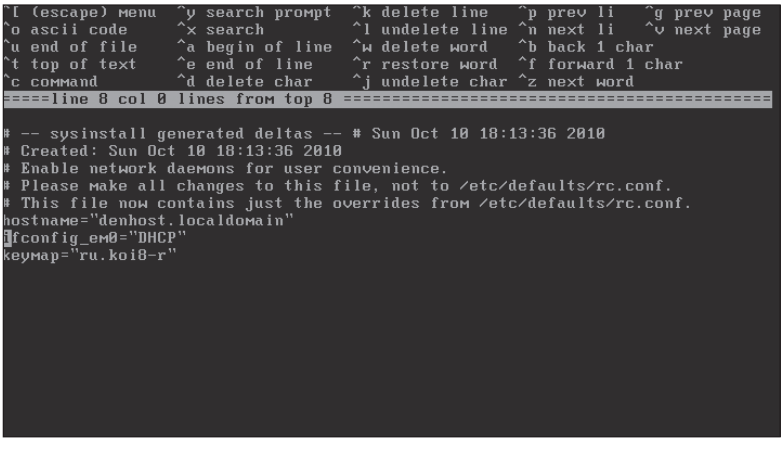

Рис. 2.18. Редактирование файла /etc/rc.conf

Для установки пароля root введите команду:

```
# passwd
```

Теперь учетная запись root защищена паролем. Осталось только создать обычного пользователя, под которым вы и будете выполнять обычные операции (которые не относятся к администрированию системы). Для создания пользователя введите команду:

```
# adduser
```

Подробно о создании пользователей мы поговорим в главе 9. А пока введите только логин пользователя, а на все остальные вопросы, кроме пароля, можно

[1] При этом ни **#**, ни **$** не являются частью самих команд, как правило, именно эти символы видны на экране как часть приглашения операционной системы к вводу команд. При этом приглашение может состоять только из них и не содержать ничего более или включать дополнительную информацию (имя текущего пользователя, путь к текущему каталогу и т. д.).

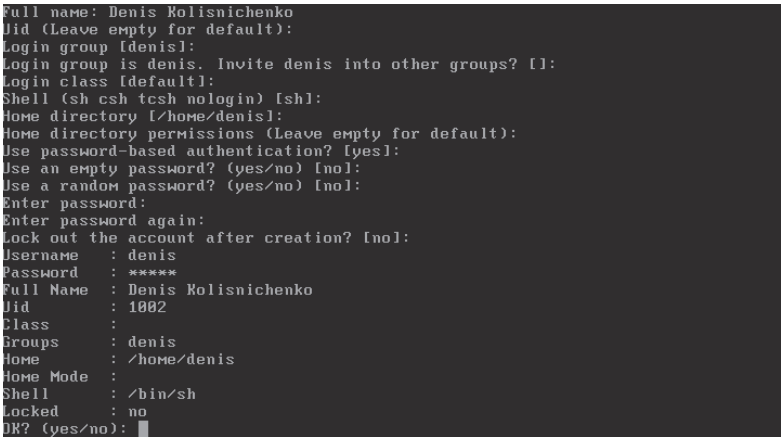

Рис. 2.19. Процесс создания пользователя

ответить, нажав Enter. Процесс создания учетной записи пользователя изображен на рис. 2.19.

Для завершения работы используйте команду `halt`, а подробно команды остановки UNIX будут рассмотрены в главе 4.

Вот на этом и все. Мы выполнили базовую настройку системы, потратив гораздо меньше времени, чем бы это заняло при выборе стандартного типа установки. Теперь можете с чистой совестью приступить к чтению следующей главы.

Резервное копирование

3

3.1. Необходимость в резервном копировании

Необходимость в создании резервных копий возникает у каждого администратора, поскольку не хочется:

- объяснять пользователям, куда пропали их файлы, хранящиеся на сервере;
- объяснять начальству, почему сеть целый день не работает (ведь сервер вышел из строя и нужно как минимум переустановить систему и настроить ее);
- тратить время на переустановку и настройку системы с нуля.

Существует множество инструментов для создания резервных копий. Каждый инструмент хорош по-своему. В простейшем случае можно просто использовать команду `cp` для копирования нужных вам данных. А что? Тоже хороший инструмент. Копия создана? Создана. Можно ее восстановить? Да проще простого — скопировать файл на «исходную позицию». К тому же у `cp` есть множество преимуществ перед другими средствами. Во-первых, команда `cp` очень быстра — не нужно тратить время на упаковку и распаковку данных. Во-вторых, для создания и восстановления файлов не нужны дополнительные программы. Инструментом `cp` пользовался каждый администратор; как правило, перед редактированием файла нужно сохранить его копию, чтобы в случае чего можно было легко вернуть все, как было.

Но у команды `cp` есть один недостаток. Она не сжимает данные. Следовательно, для создания копии вам понадобится столько места, сколько занимают исходные файлы, что не всегда удобно с экономической точки зрения. Ведь чтобы сделать копию пользовательских каталогов размером, скажем, в 200 Гбайт, вам придется приобрести еще один жесткий диск. Учитывая, что копию пользовательских данных нужно делать как можно чаще (чтобы удовлетворить примерно следующую просьбу пользователя: «В прошлую пятницу я внес неправильные

изменения в свой index.php, пожалуйста, восстановите его копию»), места вам понадобится много.

Поэтому вы будете искать другой инструмент. Можно, например, создать tar-архив всех пользовательских каталогов. Очень удобно — все будет в одном файле. К тому же архиватор tar поддерживает права доступа к файлам и каталогам, поэтому когда вы распакуете tar-архив, все файлы будут помещены на свои места с нужными правами доступа. К тому же `tar` умеет вызывать архиватор и сжимать полученный архив (опция `-z`). Например, для сжатия файлов в текущем каталоге нужно выполнить команду:

```
tar -zcfv archive.tar.gz
```

А для распаковки архива **archive.tar.gz** используется другая команда:

```
tar -zxfv archive.tar.gz
```

Для распаковки из архива только определенного каталога используется примерно следующая команда (в данном случае мы извлекаем каталог **/home/denis**):

```
tar --extract --file=archive.tgz home/denis
```

Казалось бы, у нас есть идеальный инструмент, поддерживающий и структуру каталогов, и права доступа, и сжатие файлов. Но у всего есть недостатки. Архиватор `tar`, да еще и со сжатием, довольно медленный. Попробуйте с его помощью сжать файлы и каталоги общим объемом, скажем, 400 Гбайт — вполне нормальный объем для реальной резервной копии реального сервера.

Продолжим поиски нашего идеального инструмента. Можно использовать программу `dump`, которая работает намного быстрее, к тому же можно добавить сжатие `bzip2`, что позволит сэкономить место на диске. Вот пример создания резервной копии раздела **ad0s1d**:

```
# dump -0ua -L -f- /dev/ad0s1d | bzip2> ad0s1d.img.bz2
```

Распаковать этот архив можно потом так:

```
# bunzip2 --stdout ad0s1d.img.bz2 | restore -rf -
```

Теоретически, вы можете периодически создавать дампы всех BSD-разделов на другой жесткий диск. Затем можно загрузиться с LiveCD (можно использовать команду `Fixit` меню инсталляционного диска) и восстановить по одному все разделы. Конечно, нужно будет по очереди монтировать разделы, для каждого раздела вводить команду восстановления и команду размонтирования. Хотелось бы какой-то автоматизации: загрузился, выбрал команду восстановления, выбрал хранилище резервных копий, выбрал, куда нужно восстановить, подождал немного и после перезагрузки система полностью готова к работе. Можно написать сценарии, автоматизирующие процесс резервного копирования и восстановления данных. Но зачем изобретать колесо, если все сделано до нас?

3.2. Почти идеальный инструмент для создания резервной копии и клонирования системы

Для себя я нашел почти идеальный инструмент для создания резервных копий — Clonezilla. Вообще-то Clonezilla — это инструмент не для создания резервных копий, а для клонирования системы. Суть его использования заключается в следующем. Вы устанавливаете систему и настраиваете ее. После запускаете Clonezilla и в результате получаете загрузочный диск с клоном вашей системы. Что делать с этим диском потом, зависит от поставленной задачи. Если вам нужно настроить еще 50 компьютеров аналогичной аппаратной конфигурации, можете сделать еще несколько копий «диска клонирования» и параллельно устанавливать систему. Процесс восстановления системы будет происходить намного быстрее, чем установка. Ведь при установке система спрашивает вас, что вы хотите установить, потом, исходя из вашего ответа, отбирает для установки необходимые файлы (пакеты), переносит их на жесткий диск и распаковывает. Также в процессе установки вам задается ряд вопросов, на которые нужно обдуманно ответить. А вот при восстановлении системы Clonezilla просто переносит ранее созданный образ эталонного жесткого диска на жесткий диск целевой машины. В результате на все про все уходит 10–15 минут. Быстро? Я тоже так думаю.

Даже если вам не нужно клонировать эталонную машину на несколько десятков компьютеров, Clonezilla вам пригодится. Представьте, что жесткий диск вашего сервера вышел из строя. Сколько времени понадобится на восстановление системы? Если не принимать во внимание время, затраченное на поиск нового жесткого диска, то на восстановление системы уйдут те самые 10–15 минут плюс время, необходимое на установку нового жесткого диска (вдруг у вас будет запасной!). Начальство и пользователи будут довольны.

Понятно, что только что установленная и настроенная система занимает совсем немного места, поэтому она поместится на DVD. А что делать, если у вас сервер баз данных и сама база данных разрослась, скажем, до 100 Гбайт? Clonezilla и тут поможет — вы можете хранить образы резервной копии на другом жестком диске (или на этом же, но как тогда вам поможет резервная копия, если выйдет из строя жесткий диск с резервной копией?). Правда, в этом случае использовать Clonezilla не очень удобно, поскольку она инициирует полное восстановление системы. Но зачем нам восстанавливать всю систему, если нужно восстановить всего один файл? Так что Clonezilla поможет вам при полном крахе системы, а вот файлы пользователей и системные файлы конфигурации (каталоги /etc и /usr/local/etc) нужно будет копировать с помощью `tar` или `dump`.

Программа Clonezilla может использоваться не только для клонирования FreeBSD, но также Linux и Windows. Обычно предприятие закупает компьютеры одинаковой конфигурации, но в процессе постепенного перехода на свободное бесплатное программное обеспечение часть компьютеров будет работать под Windows, часть — под Linux, поэтому можно сказать, что Clonezilla — универсальный инструмент для администратора. К примеру, нужно переустановить Windows со всеми программами. Вы копируете на флешку документы пользователя, затем

загружаетесь с диска восстановления, который вы предварительно создали, и через 15 минут получаете полностью готовый компьютер с установленной Windows и всеми программами.

Давайте разберемся, как использовать Clonezilla. Скачайте с http://clonezilla.org ISO-образ Clonezilla и запишите его на болванку. Загрузитесь с него. Сначала нужно будет выбрать язык и клавиатуру, но поскольку русского интерфейса для Clonezilla нет, оставьте все по умолчанию. Затем нужно выбрать команду `Start Clonezila` (рис. 3.1).

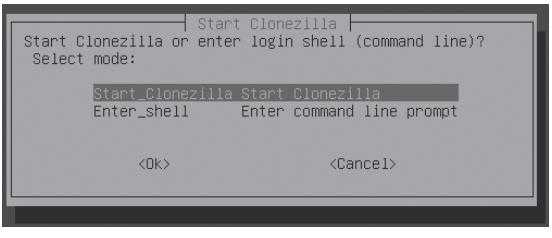

Рис. 3.1. Запустить Clonezilla

Далее нужно выбрать режим работы программы (рис. 3.2):

- device-image — создание образа раздела;
- device-device — резервная копия будет помещена на другой раздел (используется при большом размере раздела).

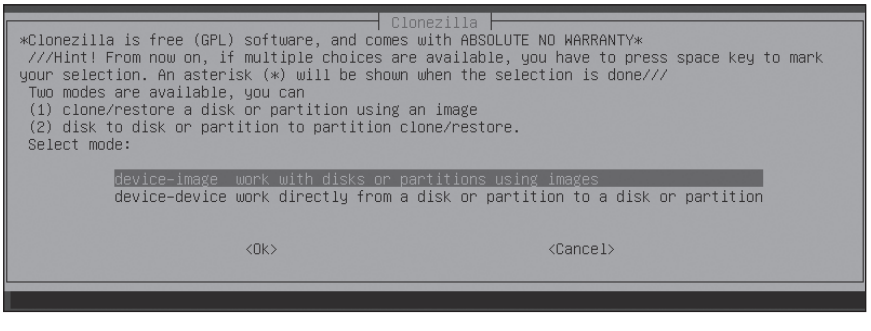

Рис. 3.2. Выбор режима работы

Следующий шаг — выбор типа хранилища резервной копии. Выберите local_dev (рис. 3.3), что означает запись резервной копии на локальное устройство (в случае создания резервной копии) или чтение резервной копии с локального устройства (в случае восстановления системы). Сразу после этого нужно будет выбрать раздел-хранилище.

После этого нужно выбрать режим работы программы (рис. 3.4):

- savedisk — сохранить весь жесткий диск как образ;
- saveparts — сохранить отдельно выбранный раздел жесткого диска (речь идет о разделах жесткого диска, а не BSD-разделах);

- restoredisk — восстановить весь диск из резервной копии;
- restoreparts — восстановить раздел из резервной копии;
- recovery-iso-zip — создать загрузочный диск восстановления (клонирования) системы;
- exit — выход.

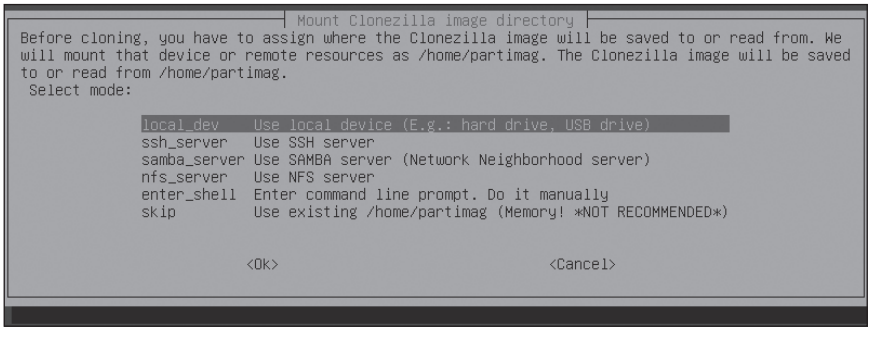

Рис. 3.3. Выбор типа хранилища

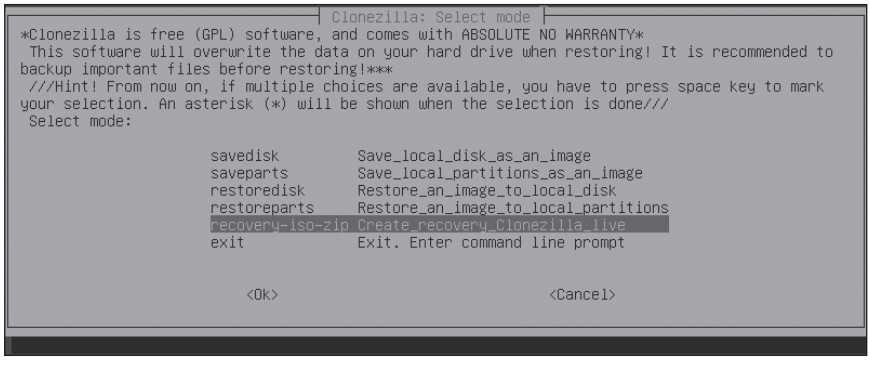

Рис. 3.4. Режим работы программы

Дальше вы справитесь сами. Хочу только отметить следующее: когда программа запросит ввести имя устройства, нужно вводить имя устройства в стиле Linux, то есть sda1, а не ad0s1.

Принцип работы Clonezilla, наверное, напомнил опытным Windows-администраторам популярный продукт Norton Ghost. Однако, в отличие от «привидения Нортона», Clonezilla — абсолютно бесплатная программа. Так что в этой главе мы познакомились еще с одним бесплатным аналогом популярного коммерческого продукта. Кстати, сейчас бесплатная Clonezilla сэкономила вам 69 долларов (на самом деле чуть больше, учитывая банковские услуги) — именно за столько продается Norton Ghost для Windows 7:

http://us.norton.com/ghost/

В заключение этой главы настоятельно рекомендую прочитать страницы справочного руководства по командам `tar`, `dump` и `restore` — эти команды вы также будете использовать при создании резервных копий. Можно также ознакомиться с архиваторами `bzip2` и `bunzip2`, которые мы использовали при создании и восстановлении дампа системы командами `dump` и `restore`.

Вход в систему и завершение работы

4

4.1. Вход в систему

В этой небольшой главе мы поговорим о том, как войти в систему после ее установки и как правильно завершить работу. Начнем с FreeBSD. Обычно на FreeBSD не устанавливается графический интерфейс. Такая возможность есть, но большинство администраторов предпочитают работать в консоли (текстовом режиме). И это решение абсолютно правильное. Незачем тратить время на настройку видеокарты (а во FreeBSD время потратить придется — это не Linux, где для корректной работы видеоподсистемы практически ничего не нужно делать), если вам нужно отредактировать несколько конфигурационных файлов, запустить сервисы и забыть вообще о существовании вашего сервера. А все административные действия можно выполнять и в консоли.

После запуска FreeBSD вы увидите приглашение войти в систему:

```
FreeBSD/i386 (den.dkws.org.ua) (ttyv0)
login: root
Password:
```

Введите имя пользователя (root) и его пароль. При вводе символы пароля никак не отображаются — не будет ни звездочек, ни точек, — просто введите пароль и нажмите Enter. Вы вошли в систему. Что делать дальше? А дальше вам нужно вводить различные команды в зависимости от того, что вы хотите сделать. Желаете отредактировать конфигурационный файл? Запустите текстовый редактор и откройте необходимый файл. Командной строке посвящена глава 6. В ней вы узнаете, как выбрать командный интерпретатор, как его использовать и настроить, и, конечно же, познакомитесь с самыми полезными командами UNIX.

А мы продолжаем изучение входа в систему. Настала очередь Linux. Если при установке системы вы выбрали запуск в текстовом режиме (на третьем уровне запуска — см. главу 8), то приглашение будет практически таким же, как и в случае с FreeBSD, только в первой строчке будет отображено название дистрибутива Linux:

```
Fedora release 13 (Goddard)
Kernel 2.6.33.3-85.fc13.i686.PAE on an i686 (tty1)
localhost login:
```

Приглашение Linux более информативно. Давайте сравним приглашение FreeBSD и Linux. В первой строчке FreeBSD выводит: название системы (FreeBSD), архитектуру, под которую оптимизирована система (i386), имя узла (den.dkws.org.ua) и имя виртуальной консоли (ttyv0):

```
FreeBSD/i386 (den.dkws.org.ua) (ttyv0)
```

Переключение между виртуальными консолями осуществляется с помощью нажатия Alt + Fn, n — число. Нумерация консолей в FreeBSD начинается с 0, а в Linux с 1. Поэтому чтобы попасть на консоль ttyv1, в Linux нужно нажать Alt + F1, а в FreeBSD — Alt + F2. Нажатие Alt + F1 в FreeBSD переключит вас на консоль ttyv0. Подробно о назначении консолей, о русификации консоли мы поговорим в главе 6.

Первая строка приглашения в Linux сообщает нам только название (Fedora), версию (13) и кодовое имя (Goddard) дистрибутива:

```
Fedora release 13 (Goddard)
```

Зато вторая строка гораздо более информативна. Она сообщает нам версию ядра (2.6.33.3-85), дистрибутив, для которого было собрано ядро (fc13), архитектуру ядра (i686); режим процессора, под который оптимизировано ядро (PAE). После этого идет указание реальной архитектуры, на которой запущена система (i686), и имя виртуальной консоли (tty1). Как видите, имена консолей в FreeBSD (ttyvN) и Linux (ttyN) немного отличаются. Вот та самая вторая строчка, чтобы вы не листали книгу:

```
Kernel 2.6.33.3-85.fc13.i686.PAE on an i686 (tty1)
```

Нужно сделать несколько замечаний. Во-первых, архитектура, под которую оптимизировано ядро, не всегда может совпадать с архитектурой, на которой запущена система. В нашем случае все совпало. Но бывает, что ядро оптимизировано под i386 (для большей совместимости), а система запущена реально на i686.

PAE (Physical Address Extension) — режим работы блока управления памятью x86-совместимых процессоров. В этом режиме используются 64-битные элементы таблицы страниц (хотя на самом деле используются первые 36 бит), с помощью которых процессор может адресовать 64 Гбайт физической памяти. А если бы использовались 32-разрядные таблицы, то процессор мог бы адресовать всего 4 Гбайт «оперативки». В этом и заключается ответ, почему, если вы установили Linux на ваш новый и ультрасовременный сервер с 16 Гбайт оперативной памяти, Linux «видит» всего 4 Гбайт. В таком не очень приятном случае вам нужно либо перекомпилировать ядро, включив поддержку PAE, либо установить другой дистрибутив, где поддержка PAE в ядре включена по умолчанию. Fedora 13 как раз является таким дистрибутивом.

Далее FreeBSD просит ввести имя пользователя:

```
login:
```

В Linux приглашение ввода имени пользователя выглядит иначе:

```
localhost login:
```

В этом случае localhost — это имя узла, на котором запущена Linux. FreeBSD выводила имя узла в первой строчке, а Linux выводит в третьей. При локальной

регистрации имя узла не имеет никакого значения, но при удаленной регистрации, особенно когда в вашей сети несколько машин под управлением FreeBSD или Linux, имя узла позволяет узнать машину, к которой вы хотите подключиться.

Большинство современных дистрибутивов Linux по умолчанию настроены на графический вход в систему. На рис. 4.1 изображено окно входа в систему дистрибутива Fedora 13. Сначала вам нужно выбрать пользователя, затем появится поле ввода пароля, как и показано на рис. 4.1. То, что кнопка Отменить не поместилась в окне, виноват не я, а разработчики Fedora. Именно поэтому я рекомендую использовать его в качестве серверного дистрибутива. На сервере все равно, как будут выглядеть кнопки, все равно, если некоторые кнопки не переведены на русский язык. А пользователям это не нравится, особенно после «вылизанной» в эстетическом плане Windows. В нижней части окна имеются списки выбора языка и раскладки клавиатуры. Если бы была установлена еще одна графическая среда, например KDE (я обычно устанавливаю одну графическую среду — GNOME), то появилась бы возможность выбрать сеанс (графическую среду). Кнопка в нижнем правом углу позволяет выключить или перезагрузить компьютер без входа в систему.

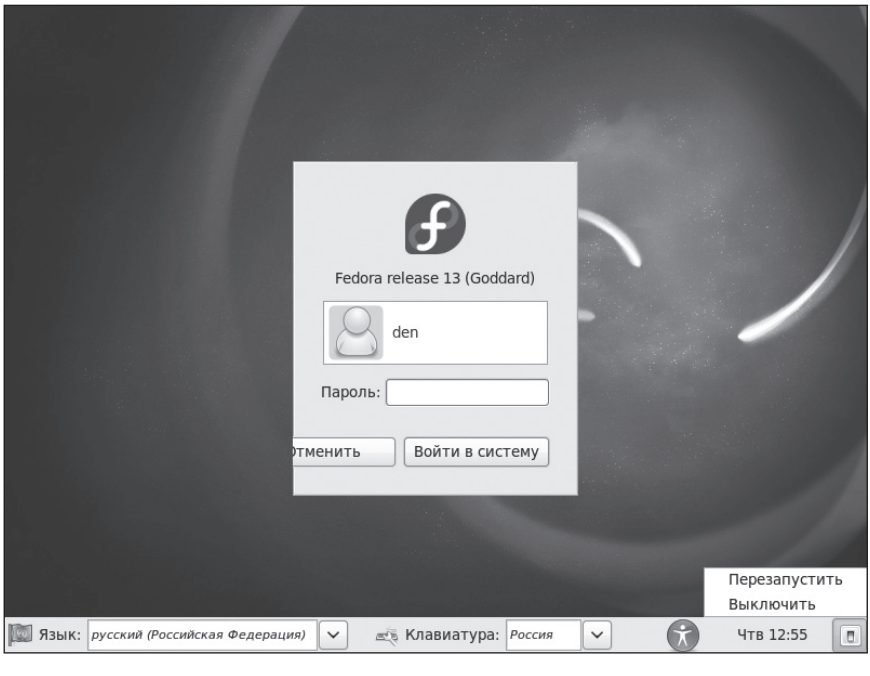

Рис. 4.1. Окно регистрации в графическом режиме

Взгляните на окно регистрации дистрибутива openSUSE 11.3 (рис. 4.2). Правда, красивее? Кстати, кнопка Тип сеанса позволяет выбрать графическую среду (рис. 4.3)

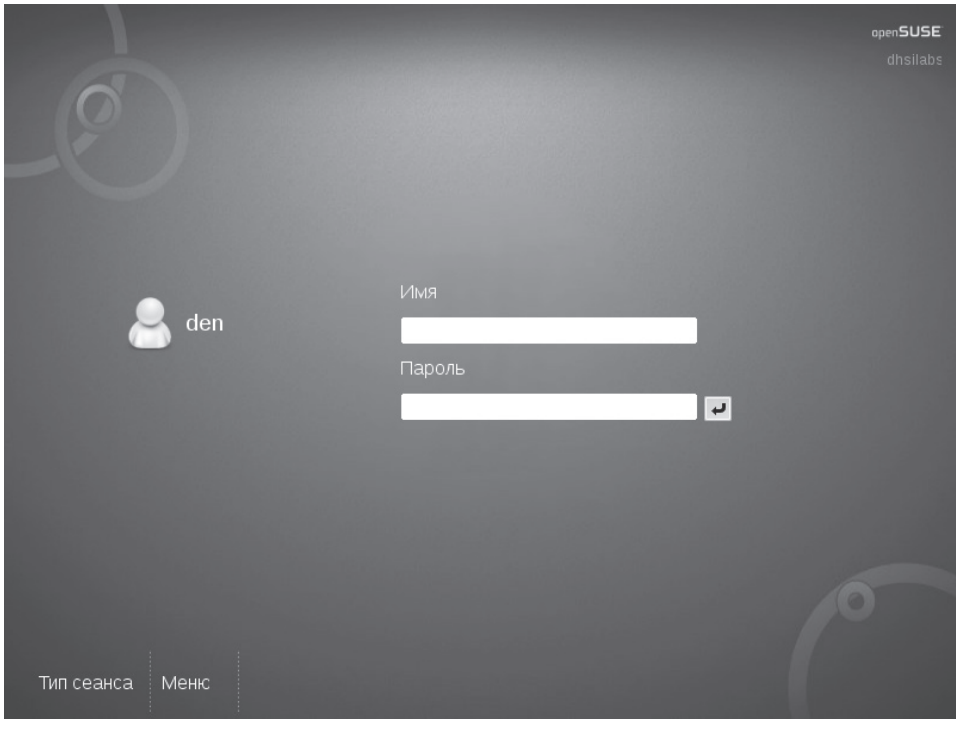

Рис. 4.2. Выход в систему (openSUSE)

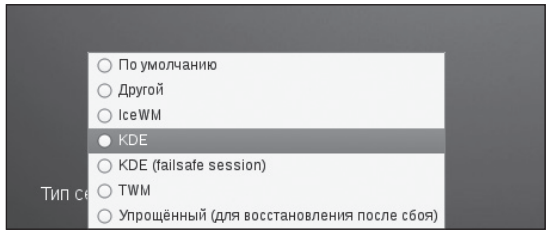

Рис. 4.3. Выбор графического сеанса (openSUSE)

Разработчики openSUSE уделяют больше внимание мелочам, именно поэтому их дистрибутив подходит как для рабочей станции, так и для сервера.

Даже если Linux загрузилась в графическом режиме, вы можете войти в консоль, нажав Ctrl + Alt + F1. Все: теперь вы в текстовом режиме, можете войти в систему и приступить к вводу команд. Хотя если вы новичок, вам будет привычнее оставаться в «графике». Но имейте в виду: чтобы стать настоящим администратором UNIX, переход в консоль неизбежен. Ведь далеко не все административные приложения имеют графическую оболочку, поэтому даже в графическом режиме вам придется вводить команды — в эмуляторе терминала. Но об этом мы поговорим в следующей главе.

4.2. Завершение работы

UNIX, как и любая другая многозадачная операционная система, нуждается в правильном завершении работы. Нельзя просто выдернуть шнур из розетки, иначе при следующем включении, даже если система и запустится, потери данных неизбежны. Ведь нет никакой гарантии, что в момент отключения питания ни одна служба не сохраняла данные. Казалось бы, все, что я здесь пишу, очевидно и понятно. Да, для любого нормального человека. Но встречаются и ненормальные. Помню, как один администратор выключал сервер путем отключения сетевого фильтра («пилота»), зато корректно завершал работу Windows 98 (в те времена она была основной операционной системой). Попытался объяснить, что так делать не нужно. Он никак не мог понять меня. Он думал, что раз у системы нет графического интерфейса, то можно просто выключить питание. Но ведь видеокарта — это не мотор с турбонаддувом. Корректное завершение работы производится не для того, чтобы турбина на видеокарте успела остыть, а чтобы все приложения (и сетевые службы в том числе) успели сохранить данные на диск, чтобы эти данные были физически записаны на диск, чтобы припарковались головки жесткого диска и т. д.

Администраторам UNIX нужно запомнить одну универсальную команду завершения работы: `shutdown`. Формат ее вызова следующий:

```
shutdown [-] [действие] время [сообщение]
```

Данную команду можно вводить как в FreeBSD, так и в Linux. Но помните, что ее нужно вводить только от имени root. Если вы работаете под именем обычного пользователя, то сначала вам нужно ввести команду `su`, затем пароль пользователя root, а затем уже — команду `shutdown`. Настоятельно рекомендую внимательно прочитать главу 9, в которой будут рассмотрены все нюансы использования команды `su`. Пока лишь скажу, что команду `su` имеют право выполнять не все пользователи, а только члены группы `wheel`. О том, как добавить пользователя в эту группу, будет показано в главе 9.

В табл. 4.1 указаны действия команды `shutdown`.

Таблица 4.1. Действия команды shutdown[1]

Действие	Описание
-h	Завершить работу системы в указанное время, перед завершением работы всем пользователям отправить указанное сообщение
-r	Перезагрузить систему в указанное время, перед завершением работы всем пользователям отправить указанное сообщение
-p	Завершить работу системы в указанное время, выключить питание компьютера. О сообщении, думаю, еще раз вспоминать не стоит — оно также будет показано

[1] Настоятельно рекомендуется уточнить (например, с помощью команды man shutdown) перечень и синтаксис действий данной команды в вашем дистрибутиве — например, в Ubuntu действий больше, а для выключения необходимо указание –P (прописная латинская буква P). Также необходимо уточнить, имеются ли в вашем дистрибутиве все сокращенные команды, указанные после таблицы.

Действие	Описание
-k	Действие «выбрасывает» всех зарегистрированных в системе пользователей. Полезно, если вы подозреваете, что одна из учетных записей скомпрометирована. До перезагрузки системы регистрация в системе будет запрещена, то есть пользователи не смогут войти в систему

Последний параметр — сообщение — можно не указывать. Если же вы не хотите указывать его в командной строке, а хотите ввести с клавиатуры, укажите параметр - перед действием:

```
shutdown - -h now
```

Время указывается в формате ЧЧ:ММ, но для немедленного завершения работы (перезагрузки) можно указать значение now, как в только что приведенной команде. Вот несколько примеров использования команды shutdown:

```
shutdown -h 20:00 "Have a nice day!"
shutdown -r now
shutdown -k now "Registration in system is forbidden"
```

Если вам лень вводить длинную команду shutdown, тогда специально для вас созданы следующие команды:

- halt — остановить работу системы (обычное завершение работы);
- poweroff — завершить работу и отключить питание;
- reboot — перезагрузить систему.

Все эти команды нужно вводить от имени root. Давайте сразу договоримся, что если команду нужно ввести от имени root, то перед ней мы будем указывать решетку (#), а если команду можно ввести от имени обычного пользователя — то доллар ($) или вообще ничего не будем указывать. Примеры:

```
# reboot
$ df
free
```

Это не наш маленький секрет, так принято в мире UNIX. Войдите в систему как обычный пользователь и посмотрите на приглашение командной строки. В нем может быть имя пользователя, домашний каталог и прочие полезности (вид приглашения можно настроить под себя — см. главу 6). Но приглашение обязательно будет заканчиваться символом доллара ($). А вот если вы войдете как root, то приглашение будет заканчиваться решеткой. А в некоторых системах вы вообще, кроме решетки, ничего не увидите: так настроено приглашение.

Вернемся к завершению работы. Выключать питание нужно не сразу, а только когда увидите соответствующее сообщение системы. Например, в случае с FreeBSD система сообщит о возможности отключения питания так:

```
The operating system has halted.
Please press any key to reboot.
```

Вот сейчас можно отключить питание компьютера. Хотя можно сразу ввести команду poweroff, и питание будет выключено автоматически. Второе сообщение говорит о возможности нажатия любой клавиши для перезагрузки компьютера.

Пользователям Linux, выбравшим графический режим, можно порекомендовать выключение питания с помощью меню GNOME/KDE. Там все просто: если вы разобрались с Windows, то с GNOME/KDE уж точно справитесь. На рис. 4.4 изображена команда завершения работы в графической среде GNOME.

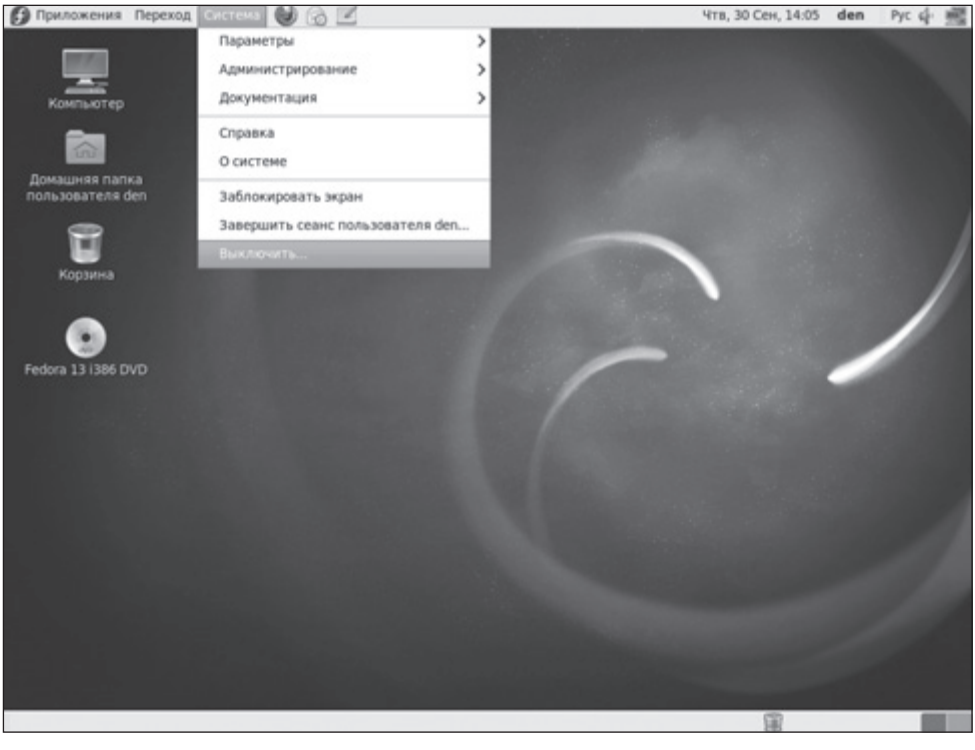

Рис. 4.4. Как выключить компьютер в GNOME

Графический интерфейс в UNIX

5

5.1. Что такое консоль? Эмуляторы консоли

Графическая система X Window System (сейчас называется X.Org[1]) впервые появилась в 1984 году и была доступна в некоторых UNIX-системах. Когда появилась первая версия Linux, а это было в далеком 1991 году, в ней не было графического интерфейса. В 1992 году, когда в Linux появилась поддержка сети и протокола TCP/IP, на Linux была портирована графическая система X Window.

X Window предоставляла только фундамент графического интерфейса, а «рисованием» элементов графического интерфейса (оформления окон, меню, списков и т. д.) занимался оконный менеджер (window manager). Таких менеджеров было много, но интерфейс пользователя, предоставляемый ними, был некрасивый и неудобный (по сегодняшним меркам). Хотя, если сравнивать с Windows 3.x, которая как раз существовала в то время, то графический интерфейс Linux не был таким уж отсталым.

Все поменялось в 1995 году, когда вышла Windows 95 с ее интерфейсом пользователя, который лег в основу современных ОС Microsoft. Графический интерфейс Linux выглядел архаизмом. Да и вообще пользователи Linux того времени предпочитали работать в консоли. Считалось, что графический интерфейс — от лукавого.

Но в 1996 году появилась графическая среда KDE (K Desktop Environment). Именно графическая среда, а не оконный менеджер. По сути, KDE является оконным менеджером, но с таким огромным количеством функций и возможностей, что ее принято называть именно графической средой. Интерфейс KDE ничем не уступал интерфейсу Windows 95, а, наоборот, учитывая виртуальные рабочие столы, превосходил его.

---

[1] Точнее, X.Org Server — это свободная реализация X Window System с открытым исходным кодом, используемая в настоящее время практически всеми дистрибутивами Linux, многими дистрибутивами на базе BSD и не только.

До 1999 года KDE была лучшей графической средой для Linux, а в 1999 году появилась альтернативная графическая среда — GNOME (GNU Network Object Model Environment). По началу пользователи не спешили переходить на GNOME, и KDE по-прежнему была более популярной. Тому есть две причины: все-таки KDE уже проверенная временем среда, да и локализирована она была лучше (вопросы локализации, понятно, были актуальными только для не англоязычных пользователей).

Сейчас же GNOME и KDE — полноценные конкуренты. По своим возможностям обе среды одинаковые, поэтому выбор конкретной среды — это дело вкуса. Некоторые дистрибутивы Linux содержат обе графические среды, некоторые — только одну (зависит от предпочтений разработчиков).

Современные дистрибутивы Linux по умолчанию запускаются в графическом режиме. А раньше для запуска графического интерфейса пользователям приходилось вводить команду `startx`.

Спрашивается, а куда подевалась консоль? Консоль по-прежнему доступна, и опытные пользователи Linux для настройки системы предпочитают работать именно в консоли. Для перехода в консоль нужно нажать комбинацию клавиш Ctrl + Alt + Fn, где n — номер консоли. Всего в Linux доступно 6 виртуальных консолей. Так, для перехода на первую консоль нужно нажать комбинацию клавиш Ctrl + Alt + F1, а на шестую — Ctrl + Alt + F6. Для переключения между консолями можно использовать комбинацию клавиш Alt + Fn, а для возврата в графический режим — Alt + F7.

В BSD способы переключения те же, но поскольку нумерация консолей производится с 0, то нажатие Ctrl + Alt + F1 приводит к переключению на консоль с номером 0, Ctrl + Alt + F2 — на консоль с номером 1 и т. д.

Когда вы переключитесь в консоль, вне зависимости, были ли вы уже зарегистрированы или нет, вам нужно будет заново указать имя пользователя и пароль. На рис. 5.1 изображен консольный вход в систему.

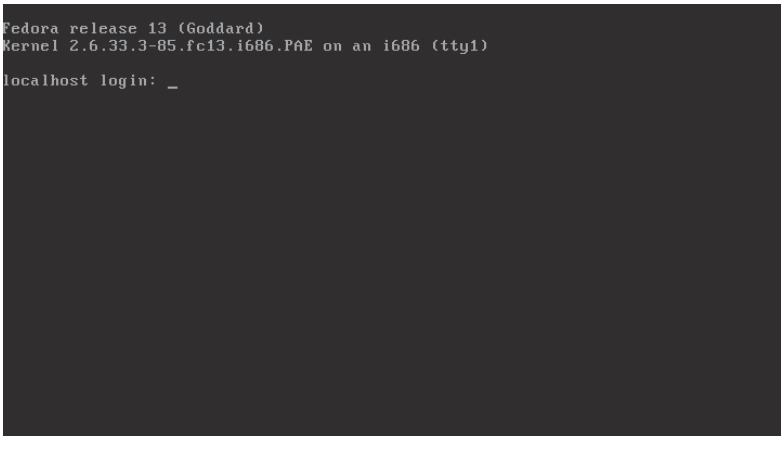

Рис. 5.1. Консольный вход в ОС Linux

Работа в консоли осуществляется путем ввода команд. Например, команда `df` выводить информацию о свободном дисковом пространстве, команда `free` — информацию об использовании оперативной памяти, а команда `top` — о запущенных процессах. В Linux есть сотни команд, и все команды вам знать совсем не обязательно. Когда вам нужно будет ввести определенную команду, я вам об этом сообщу.

Понятно, что для ввода одной-двух команд не хочется переключаться в консоль. Основная причина нежелания работать в консоли — необходимость в повторной регистрации. Да и начинающие пользователи чувствуют себя в консоли некомфортно. В таком случае намного проще использовать эмуляторы терминалов — программы, которые эмулирует поведение консоли. Для запуска такой программы выполните команду Меню ▸ Утилиты ▸ Консоль. В окне программы Консоль вы можете вводить любые команды (ввод команды, понятное дело, завершается нажатием Enter). На рис. 5.2 изображен эмулятор терминала и результат выполнения команды `free`. Из вывода команды ясно, что в системе установлено 512 Мбайт оперативной памяти, а используется системой чуть меньше половины доступной памяти.

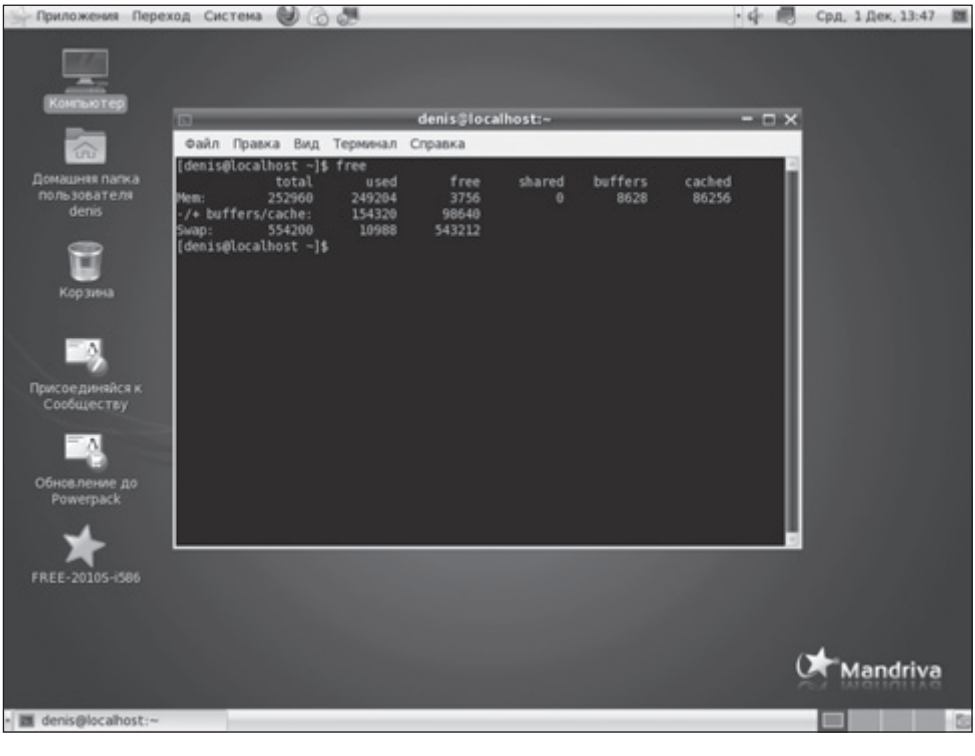

Рис. 5.2. Эмулятор терминала: 256 Мбайт вполне достаточно для запуска Mandriva 2001.1 Spring с графической средой GNOME

Иногда нам нужно быстро запустить какую-нибудь графическую программу, команду запуска которой мы знаем. Например, текстовый редактор `kwrite` или какой-нибудь конфигуратор, например `drakconf` (см. главу 3). Открывать Консоль не хочется, потому что потом вам нужно будет закрывать сразу два окна — и окно запущенной программы (когда она будет уже не нужна), и окно эмулятора консоли. В этом случае можно нажать Alt + F2 и в появившемся окне ввести нужную команду и нажать Enter (рис. 5.3).

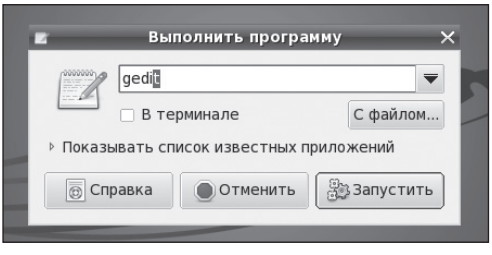

Рис. 5.3. Окно быстрого запуска программы

Такой способ неудобен для запуска текстовых программ (не графических), потому что вы не увидите результат их выполнения.

5.2. Установка графического интерфейса во FreeBSD

В Linux графический интерфейс устанавливается и настраивается автоматически. Разница может быть только в графической среде, которую вы выберете при установке. Если вы установите несколько графических менеджеров, то при входе в систему можно будет выбрать графическую среду. На рис. 5.4 показано, что можно выбрать GNOME, IceWM или drak3d для настройки трехмерных эффектов. Кстати, на более медленных компьютерах лучше выбирать среду GNOME, поскольку она менее ресурсоемка. Так, для работы Linux с графической средой GNOME вполне достаточно 256 Мбайт оперативной памяти, а вот KDE 4 (именно 4 — эта версия включена в состав современных дистрибутивов) будет подтормаживать даже на 512 Мбайт.

Во FreeBSD (да и в других BSD-системах) графический интерфейс по умолчанию не устанавливается. Его придется установить и настроить самостоятельно. В Linux мне уже более 10 лет не приходилось настраивать графический интерфейс вручную, а тем более устанавливать его. В 1999–2001 годах приходилось немного изменять параметры монитора, вроде частоты и разрешения, потому что они не всегда правильно устанавливались автоматически. Далее приходилось редактировать конфигурационный файл X-сервера, когда появились трехмерные графические интерфейсы.

В мире BSD многое приходится делать вручную. Разберемся с установкой X.Org — графической подсистемы UNIX. Далее нужно будет установить одну из графических сред — GNOME или KDE.

Рис. 5.4. Графический вход в систему в Linux

5.2.1. Установка и настройка X.Org

Первым делом нужно установить фундамент графического интерфейса — систему X.Org. Для установки X.Org введите команду:

```
# pkg_add -r xorg
```

Если не хочется загружать пакеты из Интернета, можно установить их с дистрибутивного DVD. Для этого введите команды:

```
# mount /cdrom
# cd /cdrom/packages/All
# pkg_add xorg
```

После установки X.Org вам не нужно его настраивать. Благодаря демону `hald` настройка компонентов графической подсистемы (видеокарты, монитора, клавиатуры, мыши) осуществляется автоматически. Но вам нужно обеспечить автоматический запуск этих демонов, добавив в `/etc/rc.conf` строки:

```
hald_enable="YES"
dbus_enable="YES"
```

Чтобы не перезагружать машину после этого, запустим сервисы `hald` и `dbus` вручную.

```
# /usr/local/etc/rc.d/hald start
# /usr/local/etc/rc.d/dbus start
```

При настройке графического интерфейса вы должны знать одну тонкость, которую не найдете в многочисленных руководствах по X.Org, GNOME и KDE. Для полноценной работы GNOME и KDE вам нужно подмонтировать псевдофайловую систему procfs. В Linux эта файловая система монтируется автоматически, поэтому никаких проблем с запуском GNOME и KDE быть не может, а в FreeBSD она по умолчанию отключена. Для ее включения добавьте в /etc/fstab строку:

```
proc /proc procfs rw 0 0
```

После этого выполните команду mount -a для монтирования файловой системы или перезагрузите компьютер.

Практически все уже сделано, теперь осталось запустить X.Org, чтобы проверить, запустится ли X-сервер с настройками по умолчанию. Для этого введите команды:

```
# setxkbmap -option terminate:ctrl_alt_bksp
# startx
```

Первая команда разрешает использование клавиш Ctrl + Alt + Backspace для завершения работы X-сервера, а вторая запускает X-сервер.

Вы должны увидеть два ужасных графических терминала. Они настолько ужасны, что даже не хочется делать иллюстрацию, но вы сразу поймете, о чем я говорю. Если никаких графических окон не появилось, значит, не получилось автоматически настроить X.Org.

Попробуем сгенерировать файл конфигурации вручную:

```
# Xorg -configure
```

В каталоге /root будет создан конфигурационный файл xorg.conf.new, попробуем запустить X-сервер с этим конфигурационным файлом (если что-то пойдет не так, не забывайте о Ctrl + Alt + Backspace):

```
# Xorg -config xorg.conf.new -retro
```

Если все прошло успешно, скопируйте этот файл в каталог /etc/X11:

```
# cp /root/xorg.conf.new /etc/X11/xorg.conf
```

При повторе проблемы нужно редактировать файл xorg.conf, чтобы он соответствовал вашим аппаратным средствам. Иногда проще найти xorg.conf в Интернете — вряд ли вы единственный человек с подобной видеокартой и монитором. Монитор в xorg.conf описывается так:

```
Section "Monitor"
Identifier "Monitor0"
VendorName "Monitor Vendor"
ModelName "Monitor Model"
HorizSync 30-107
VertRefresh 48-120
EndSection
```

Значения горизонтальной развертки и частоты обновления вы можете найти в документации по монитору или на сайте производителя. Видеокарта описывается так:

```
Section "Device"
Identifier "NVIDIA Corporation"
Driver "nvidia"
EndSection
```

Секция Screen связывает секции Device и Monitor воедино:

```
Section "Screen"
Identifier "Screen0"
Device "Card0"
Monitor "Monitor0"
DefaultDepth 24
SubSection "Display"
Viewport 0 0
Depth 24
Modes "1024x768"
EndSubSection
EndSection
```

Скорее всего, в Интернете есть описание секции Device дл вашей видеокарты. Некоторые примеры настройки X-сервера можно найти в руководстве по FreeBSD:

http://www.freebsd.org/doc/ru/books/handbook/x-config.html

В листинге 5.1 приводится рабочий пример файла xorg.conf для видеокарты nVidia.

Листинг 5.1. Пример файла конфигурации xorg.conf для nVidia GF7600

```
Section "Module"
    SubSection  "extmod"
    EndSubSection

EndSection

Section "Files"

    FontPath    "/usr/local/lib/X11/fonts/misc"
    FontPath    "/usr/local/lib/X11/fonts/TTF"
    FontPath    "/usr/local/lib/X11/fonts/OTF"
    FontPath    "/usr/local/lib/X11/fonts/Type1"
    FontPath    "/usr/local/lib/X11/fonts/100dpi"
    FontPath    "/usr/local/lib/X11/fonts/75dpi"
    FontPath    "/usr/local/lib/X11/fonts/local"
    FontPath    "/usr/local/lib/X11/fonts/webfonts"
    FontPath    "/usr/local/share/fonts"

EndSection

Section "InputDevice"

    Identifier  "Keyboard1"
    Driver      "kbd"
```

продолжение ⇩

Листинг 5.1 *(продолжение)*

```
    Option "AutoRepeat" "500 30"
    Option "XkbRules"   "xorg"
    Option "XkbModel"   "pc101"
    Option "XkbLayout" "us,ru(winkeys)"
    Option "XkbOptions" "grp:ctrl_shift_toggle,grp_led:scroll,altwin:menu"

EndSection

Section "InputDevice"


    Identifier  "Mouse1"
    Driver      "mouse"
    Option "Device"      "/dev/sysmouse"
    Option "ZAxisMapping"   "4 5 6 7"

EndSection

Section "Monitor"

    Identifier  "PF790"

    HorizSync   30.0 - 97.0
    VertRefresh 50-180

    Modeline "1024x768@100"  113.31  1024 1096 1208 1392  768 769 772 814  -HSync +Vsync
    Modeline "1024x768@118"  136.57  1024 1104 1216 1408  768 769 772 822  -HSync +Vsync
    Modeline "1152x864@100"  143.47  1152 1232 1360 1568  864 865 868 915  -HSync +Vsync
    Modeline "1280x1024@089"  167.32  1280 1376 1512 1744  1024 1025 1028 1078  -HSync +Vsync

EndSection

Section "Device"
    Identifier  "Standard VGA"
    VendorName  "Unknown"
    BoardName   "Unknown"
    Driver     "vga"
EndSection

Section "Device"
    Identifier  "GF7600"
    Driver      "nvidia"

EndSection

Section "Screen"
    Identifier  "Screen 1"
    Device      "GF7600"
    Monitor     "PF790"
    DefaultDepth 24

    Subsection "Display"
        Depth       8
        Modes       "1152x864@100" "1024x768@118"
        ViewPort    0 0
```

```
    EndSubsection
    Subsection "Display"
        Depth       16
        Modes       "1152x864@100" "1024x768@118"
        ViewPort    0 0
    EndSubsection
    Subsection "Display"
        Depth       24
        Modes       "1152x864@100" "1024x768@118"
        ViewPort    0 0
    EndSubsection
EndSection

Section "ServerLayout"

    Identifier  "Simple Layout"
    Screen "Screen 1"
    InputDevice "Mouse1" "CorePointer"
    InputDevice "Keyboard1" "CoreKeyboard"

EndSection
```

Для работы этого примера нужно установить порт nvidia-driver, содержащий драйвер nvidia:

```
# cd /usr/ports/x11/nvidia-driver
# make install clean
```

5.2.2. Установка шрифтов

Для нормальной работы GNOME и KDE желательно установить дополнительные шрифты (иначе графический интерфейс не будет выглядеть привлекательно):

```
# cd /usr/ports/x11-fonts/urwfonts
# make install clean
```

После установки порта нужно добавить в xorg.conf строку:

```
FontPath "/usr/local/lib/X11/fonts/URW/"
```

Следующий шаг — включение TrueType, для этого в секцию Monitor конфигурационного файла нужно добавить строку:

```
Load "freetype"
```

Затем создайте каталог /usr/local/lib/X11/fonts/TrueType и скопируйте в него шрифты с MS Windows. После этого добавьте строку FontPath:

```
FontPath "/usr/local/lib/X11/fonts/TrueType"
```

Вот теперь можно приступить к установке GNOME и KDE.

5.2.3. Установка GNOME

Для установки GNOME введите следующие команды:

```
# pkg_add -r gnome2
# pkg_add -r gnome2-fifth-toe
# pkg_add -r gnome2-power-tools
```

```
# pkg_add -r gnome2-office
# pkg_add -r gnome2-hacker-tools
```

Первая команда установит саму среду GNOME. В принципе, если вам кроме нее больше ничего не нужно, можете не устанавливать остальные пакеты. Вторая команда устанавливает дополнительные приложения для GNOME. Третья — про-

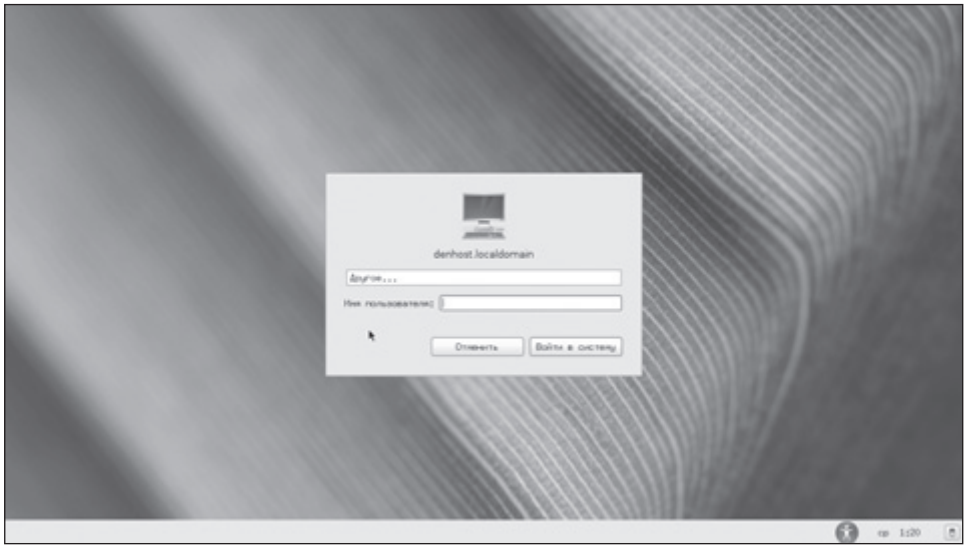

Рис. 5.5. Окно входа в систему (менеджер дисплея GDM)

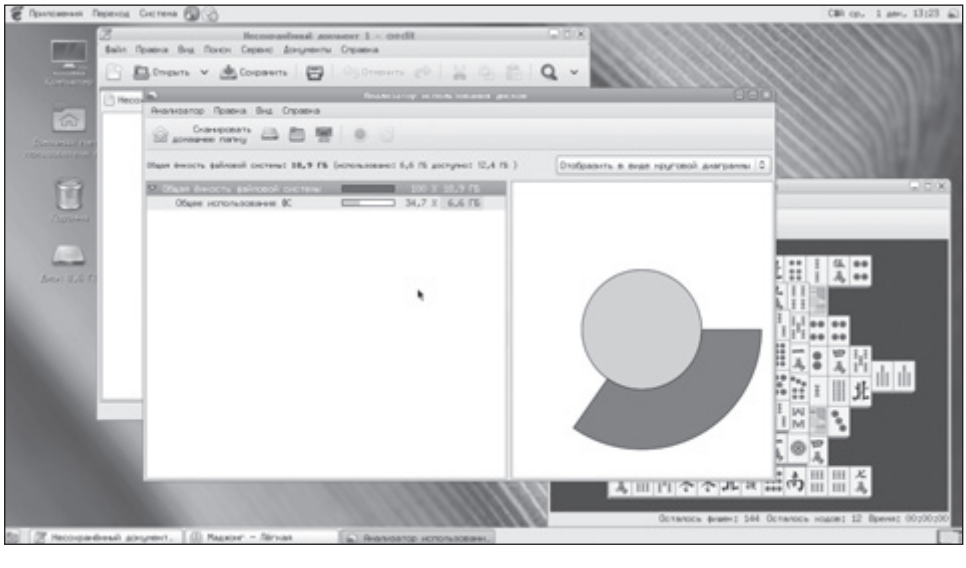

Рис. 5.6. Графическая среда GNOME во FreeBSD

граммы для администратора. Четвертая — офисные приложения, а пятая — приложения для разработчика (а не для «хакера», как можно подумать из названия пакета, хотя тут все зависит от определения «хакер»).

Чтобы GNOME запускалась при вводе команды `startx`, нужно выполнить команду:

```
echo "/usr/local/bin/gnome-session" > ~/.xinitrc
```

Данная команда добавить строку запуска gnome-session в файл .xinitrc[1]. Такую несложную манипуляцию нужно произвести для каждого пользователя вашей системы. Если пользователей много, то это не очень удобно, поэтому можно настроить автоматический запуск графической системы при запуске, как это сделано в Linux. Для этого нужно добавить следующие строки в rc.conf:

```
gdm_enable="YES"
gdm_lang="ru_RU.KOI8-R"
```

Эти строчки нужно добавить после строки запуска демона `dbus`. После этого перезагрузите машину. После перезагрузки вы увидите окно входа в систему (рис. 5.5). На рис. 5.6 изображена графическая среда GNOME в действии.

5.2.4. Графическая среда KDE

Теперь проделаем то же самое, но с KDE. Установим саму KDE:

```
# pkg_add -r kde4
```

Команда устанавливает графическую среду KDE 4. На слабых компьютерах можно установить графическую среду KDE 3:

```
# pkg_add -r kde3
```

После установки KDE введите команду:

```
echo "exec startkde" > ~/.xinitrc
```

Как и в случае с GNOME, эта команда обеспечит запуск KDE при вводе команды `startx`. Можно обеспечить загрузку KDM (K Display Manager) при старте системы. Для этого откройте /etc/ttys и отредактируйте строку, относящуюся к консоли ttyv8:

```
// Для KDE4
ttyv8 "/usr/local/kde4/bin/kdm -nodaemon" xterm on secure
// Для KDE3
ttyv8 "/usr/local/bin/kdm -nodaemon" xterm on secure
```

На рис. 5.7 изображена графическая среда KDE 4 под управлением FreeBSD.

[1] Следует уточнить, что эта команда не добавляет строку в файл, а создает файл (или полностью перезаписывает, если он существует) и только после этого записывает в него указанную строку. Для добавления строки в уже существующий файл (что в данном случае маловероятно) вместо > следует использовать >> (без пробелов между символами).

Рис. 5.7. Графическая среда KDE 4 во FreBSD

Командная строка. Полезные команды

6

6.1. Виртуальные консоли

Как уже отмечалось, для переключения между виртуальными консолями используются клавиши Alt + Fn, где n — это число. Зачем нужно несколько виртуальных консолей? Во-первых, так удобнее. На одной консоли вы можете запустить файловый менеджер mc (Midnight Commander) и просматривать файловую систему, а на второй — текстовый редактор и редактировать конфигурационный файл. На третьей консоли у вас может быть запущен сценарий мониторинга журналов. Это не вымышленная, а вполне реальная ситуация. Когда у вас появится опыт администрирования UNIX-серверов, по-другому вы даже не сможете работать. К тому же на разных консолях можно войти под разными пользователями. Например, для редактирования конфигурационных файлов нужно зайти под root, а вот для просмотра журналов системы не всегда нужны права root.

Но не нужно думать, что в консоли можно только просматривать файловую систему, вводить команды. Существуют консольные версии браузеров, почтовых клиентов, мультимедиа-проигрывателей и даже ICQ-клиентов. Так что вы можете организовать в консоли полноценное рабочее место. В табл. 6.1 приводятся примеры текстовых программ для полноценной работы в консоли.

Таблица 6.1. Примеры текстовых программ

Тип программы	Программы
Файловый менеджер	mc
Браузер	links, lynx
Почтовый клиент	mutt, pine
ICQ-клиент	centericq, finch, micq
Мультимедиа-проигрыватель	moc (Music On Console)

Вернемся к переключению консолей. Во FreeBSD нумерация консолей начинается с 0, а в Linux — с 1, хотя переключение осуществляется с помощью все тех же комбинаций клавиш Alt + Fn. В табл. 6.2 и 6.3 приводятся комбинации клавиш для переключения между консолями в Linux и FreeBSD.

Таблица 6.2. Переключение между консолями в Linux

Консоль	Комбинация	Описание
ttyN	Alt + FN	Переключение на виртуальную консоль, номер консоли соответствует числу N. Первые шесть консолей (tty1…tty6) используются для командной строки
tty7	Alt + F7	Переключение из консоли в графический режим. Обычно седьмая консоль используется для X.Org
ttyN	Ctrl + Alt + FN	Переключение из графического режима на консоль с номером N

Таблица 6.3. Переключение между консолями в FreeBSD

Консоль	Комбинация	Описание
ttyvN	Alt + FM	Поскольку нумерация консолей начинается с 0, то числа N и M — разные. Так, комбинация клавиш Alt + F1 используется для переключения на консоль ttv0, Alt + F2 — на консоль ttv1
ttyv5	Alt + F6	Используются для переключения в графический режим, если X.Org запущена
ttyv6	Alt + F7	
ttyv7	Alt + F8	
ttyvN	Ctrl + Alt + FM	Используется для переключения из графического режима на консоль

6.2. Выбор командной оболочки

6.2.1. Файл /etc/shells

Алгоритм работы в командной строке следующий: вы переключаетесь на нужную вам консоль (что мы уже научились делать), регистрируетесь в системе (вводите имя пользователя и пароль), затем в вашем распоряжении командный интерпретатор (командная оболочка). Но и здесь UNIX предоставляет свободу выбора. Как правило, вам доступно несколько командных интерпретаторов.

Список установленных командных оболочек содержится в файле /etc/shells. В листинге 6.1 приводится типичный пример этого файла для FreeBSD, а в листинге 6.2 — для Linux.

ПРИМЕЧАНИЕ

Понимаю, что пока вы еще не знакомы с командной строкой, поэтому для просмотра файла /etc/shells используйте команду cat /etc/shells.

Листинг 6.1. Файл /etc/shells в FreeBSD

```
# $FreeBSD: src/etc/shells,v 1.5.36.1 2009/08/03 08:13:06 kensmith Exp $
#
# List of acceptable shells for chpass(1).
# Ftpd will not allow users to connect who are not using
# one of these shells.
```

```
/bin/sh
/bin/csh
/bin/tcsh
/usr/local/bin/bash
/usr/local/bin/rbash
```

Листинг 6.2. Файл /etc/shells в Linux

```
/bin/ash
/bin/bash
/bin/csh
/bin/false
/bin/ksh
/bin/sh
/bin/tcsh
/bin/true
/bin/zsh
/usr/bin/csh
/usr/bin/ksh
/usr/bin/bash
/usr/bin/tcsh
/usr/bin/zsh
```

На первый взгляд в Linux установлено гораздо больше оболочек, но на самом деле программы /bin/true, /bin/false не являются оболочками (их используют при написании сценариев, а также для запрещения пользователю входа в систему, об этом мы поговорим позже). В данном примере в Linux установлены оболочки ash, bash, tcsh, ksh и zsh. Файл /bin/sh обычно является ссылкой на /bin/bash, а файл /bin/csh — обычно ссылка на tcsh. Поэтому в Linux установлено всего 5 оболочек (а не 14, как может показаться), а во FreeBSD — 3 (bash, tcsh, rbash).

В файле /etc/passwd помимо информации о пользователях указывается еще и оболочка, которая будет запущена при входе пользователя в систему. Пример:

```
den:x:550:550:Denis Kolisnichenko:/home/den:/bin/bash
```

В данном примере видно, что при входе в систему пользователя den будет запущена оболочка /bin/bash. Теоретически можно запустить любую программу в качестве оболочки — хоть проигрыватель музыки или текстовый браузер. Но эта программа должна быть обязательно прописана в файле /etc/shells, иначе запуск программы будет запрещен, и пользователь не сможет войти в систему.

Программы /bin/true и /bin/false возвращают логические значения TRUE и FALSE (что соответствует числам 1 и 0). Эти программы часто используются при написании сценариев командной оболочки, но их очень удобно использовать для запрета пользователю входить в систему. Достаточно установить программу /bin/true (или /bin/false — без разницы) в качестве командной оболочки, и пользователь больше не сможет войти в систему. После входа такого пользователя в систему будет запущена программа /bin/true, которая ничего не делает — она возвращает TRUE и завершает работу. А при завершении работы командной оболочки завершается и сеанс пользователя. То есть пользователь как бы войдет в систему, но его сеанс сразу же будет завершен.

Пример:

```
den:x:550:550:Denis Kolisnichenko:/home/den:/bin/true
```

Во FreeBSD вместо /bin/true используется программа /usr/sbin/nologin:

```
news:*:8:8:News Subsystem:/:/usr/sbin/nologin
```

6.2.2. Классическая оболочка sh

Оболочка sh является самой первой оболочкой для UNIX. Данная оболочка была разработана Стивеном Борном (Steven Bourne) в 1970-х годах для системы AT&T UNIX. Второе название этой оболочки Bourne Shell. Именно поэтому следующая версия этой оболочки была названа bash — Bourne Again Shell. (И снова оболочка Борна.)

Чуть позже данная оболочка (именно sh, а не bash) была усовершенствована и вошла с состав утилит POSIX (Portable Operating System Interface for UNIX) — переносимого интерфейса операционных систем UNIX. Именно эта усовершенствованная версия используется в FreeBSD по умолчанию для обычных пользователей. Для пользователя root используется оболочка tcsh.

Для 70-х годов прошлого века оболочка sh была чуть ли не совершенством, но сейчас на фоне того же bash она выглядит пережитком прошлого. Она не очень удобна для повседневного использования, поэтому большинство пользователей выбирают bash, а в Linux интерпретатор bash используется по умолчанию. Зато sh подходит для написания сценариев; в Интернете можно найти огромное количество сценариев, написанных на языке этой командной оболочки. Да и сценарии инициализации системы часто пишутся на языке sh. Сценарий оболочки можно сравнить с пакетным файлом MS-DOS (файл с расширением .BAT), но возможности языка оболочки намного шире.

6.2.3. Оболочка csh с синтаксисом языка C

Оболочка csh (C Shell) — стандартная оболочка для BSD, именно для BSD, а не для FreeBSD, поскольку в современных версиях FreeBSD по умолчанию используется оболочка tcsh (только для пользователя root). Разработка csh началась очень давно — еще в те времена, когда институт Беркли получил первые версии UNIX. Тогда будущим создателями BSD не понравилась стандартная оболочка sh и было решено создать новую оболочку — csh.

Как можно догадаться из названия, внутренний язык оболочки очень напоминает язык C. UNIX была написана на C, поэтому совершенно справедливо, что разработчики BSD ожидали огромной популярности от csh. Но, как оказалось на практике, разработчикам больше понравилась sh — программисты предпочитали писать сценарии на sh, а не на csh.

Зато csh существенно превосходила sh для повседневного использования. Она умеет управлять заданиями, хранит историю ранее введенных команд, обладает сценариями, которые выполняются при входе в систему и выходе пользователя из оболочки (как правило, он выполняется при выходе из системы по команде exit).

В FreeBSD используется более «продвинутая» версия csh — tcsh, а файл /bin/csh — это всего лишь ссылка на /bin/tcsh.

6.2.4. И снова оболочка Bourne: bash

Самое интересное, что Стивен Борн не имеет отношения к разработке bash. Оболочка bash была разработана фондом свободного программного обеспечения (Free Software Foundation, FSF). За основу была взята оболочка sh. В результате получилась удобная оболочка, которая быстро завоевала популярность. О ее огромной популярности свидетельствует тот факт, что она используется по умолчанию во всех дистрибутивах Linux.

Командный интерпретатор bash полностью поддерживает синтаксис sh, поэтому оболочка sh часто не устанавливается, а просто создается ссылка /bin/sh на исполнимый файл bash — /bin/bash. Но это правильно для Linux. А вот во FreeBSD перед вами оригинальная оболочка sh, а не ссылка на другую оболочку. Правда, это не разработка 70-х годов, а современная оболочка, в которую вошли решения из оболочек ksh, csh и tcsh.

Оболочка bash гораздо более удобная, чем sh: вы можете редактировать командную строку, просматривать и редактировать историю команд, создавать псевдонимы команд (это очень удобно), устанавливать и использовать в своих сценариях переменные окружения. Как и у csh, у bash есть сценарии, запускающиеся при входе и выходе из системы.

У bash есть один недостаток, но он к нам никак не относится. Дело в том, что bash обычно установлен только в бесплатных системах, а вот в коммерческих версиях UNIX вы его не найдете.

6.2.5. Оболочка ksh

После того как институт Беркли разработал для своей BSD оболочку csh, программисты компании AT&T решили не отставать и создали оболочку ksh (Korn Shell) для своей версии UNIX. Разработчиком ksh является Девид Корн (David Korn).

Нельзя сказать, что ksh создавался по образу и подобию csh, но функциональность обеих оболочек примерно одинаковая: ksh тоже умеет управлять заданиями, хранит историю введенных ранее команд, позволяет создавать конфигурационный файл для оболочек, поддерживает псевдонимы команд. Хотя ksh была разработана в далеком 1986 году, она до сих пор популярна среди программистов, поскольку синтаксис ее внутреннего языка гораздо гибче, чем языка csh.

Оболочка ksh используется по умолчанию в некоторых версиях UNIX, но не во FreeBSD. Поскольку ksh обычно устанавливается в коммерческих версиях UNIX, из свободной FreeBSD она исключена. Во FreeBSD и Linux вы можете установить общедоступную версию ksh — pdksh.

Для повседневного использования оболочка ksh не очень удобна, начинающим пользователям можно смело порекомендовать bash, но никак не ksh.

6.2.6. Оболочка tcsh — модернизированная версия csh

Изначально оболочка tcsh разрабатывалась для операционной системы TENEX, которая очень давно работала на компьютере PDP-10 производства DEC. Именно поэтому в названии оболочки есть буква «t».

За основу была взята оболочка csh. Оболочка была существенно переработана: добавлено автозавершение команд (позже мы познакомимся с этой функций), модернизирована функция редактирования командной строки и многое другое. Благодаря этому оболочка стала намного удобнее, чем ее прообраз.

Вот только с программированием опять не сложилось. Многие программисты предпочитают использовать для написания сценариев именно bash, но никак не tcsh.

6.2.7. Оболочка zsh

Оболочка zsh вызывает неоднозначные эмоции. С одной стороны, она нерасторопна и за это нельзя ее похвалить. Ведь ее создатели хотели реализовать в ней все самые «вкусные» функции ksh, bash и tcsh. А с другой стороны, она очень удобна в использовании. Пока могу сказать следующее: оболочка, насколько мне известно, не используется по умолчанию ни в одной UNIX-системе, но некоторые пользователи предпочитают именно ее. Заинтересовавшиеся читатели могут обратить внимание на интересную статью об этой оболочке:

http://opennet.ru/base/dev/zsh_intro.txt.html

6.2.8. Простая оболочка ash

Оболочку ash вы не встретите в FreeBSD, но зато она присутствует во многих дистрибутивах Linux по умолчанию. Это самая компактная и простая оболочка для UNIX. Данная оболочка обладает всего двадцатью четырьмя встроенными командами. Часто ash вызывается при загрузке Linux в однопользовательском режиме.

Частично ash совместима с sh, с ее помощью можно запускать сценарии, написанные на традиционном языке sh (а не на bash). В операционной системе NetBSD ash используется вместо /bin/sh.

6.2.9. Что выбрать?

Итак, перед нами проблема выбора. Выбирать оболочку нужно исходя из поставленных задач. Например, если вы выбираете оболочку для повседневного использования, попробуйте bash или sh/tcsh. Также советую попробовать zsh — очень удобная, хотя и не очень быстрая оболочка. Категорически не рекомендую для повседневного использования выбирать ksh и ash.

А вот для программирования оптимальным вариантом будет оболочка bash. Синтаксис написания сценариев с ее помощью мы рассмотрим в главе 34.

6.3. Использование и настройка командной оболочки

6.3.1. Основные принципы работы в командной строке. Автозавершение команд

В этой главе мы рассмотрим настройку и использование трех основных оболочек — sh, tcsh (используется по умолчанию в FreeBSD) и bash (по умолчанию в Linux).

Использовать консоль очень просто — введите команду и нажмите Enter. Команда содержит имя запускаемой программы и, если нужно, передаваемые программе параметры. Например, когда вы запускаете текстовый редактор, то, очевидно, укажете имя файла, который вам нужно отредактировать:

```
nano /etc/groups
```

В данном случае nano /etc/groups является командой, nano — имя запускаемого текстового редактора, а /etc/groups — параметр, передаваемый программе nano.

Кроме параметров команда может содержать символы перенаправления ввода/вывода (|, <, >, >>), но это тема для отдельного разговора, поэтому отложим ее на некоторое время.

Вам нужно знать, какую команду вводить. Чуть позже будут рассмотрены самые полезные команды UNIX. Если вы забыли, как называется команда, но помните начальные символы, можете использовать автозавершение командной строки. Для этого введите хотя бы первый символ программы и нажмите Tab (при использовании bash) или Ctrl + D (если у вас sh или tcsh). Интерпретатор предложит вам доступные варианты команд (если вариантов будет несколько) или сразу дополнит команду (если есть всего один вариант). Функция очень удобная.

Бывает и такое, что вы знаете команду, но не помните, что она делает, или же забыли, что означает тот или иной параметр, используйте команду man <имя_программы>. Команда man отображает справочную страницу по указанной программе. Если что-то не получается, не забывайте о man! В большинстве случаев, как показывает практика, справочная система содержит ответы на 90% возникающих вопросов. Но почему-то многие администраторы пренебрегают ею и потом задают ненужные вопросы на форумах. Раньше читать справочные страницы было не очень удобно, поскольку они были на английском языке, сейчас можно установить пакет, содержащий справку на русском языке, по крайней мере, для основных команд. Также можно ввести запрос в Google вида: man <программа>. Google довольно интеллектуальный поисковик, поэтому сразу найдет страницу руководства, причем, по возможности, на русском языке.

Для просмотра истории команд (списка ранее введенных команд) используйте стрелки ↑ и ↓. Нашли нужную команду? Тогда нажмите Enter, и команда будет выполнена. Список последних команд хранится в файле .history (если у вас tcsh) или .bash_history (если у вас bash).

6.3.2. Перенаправление ввода/вывода

С помощью перенаправления ввода/вывода можно перенаправить вывод одной команды на ввод другой или в файл. Предположим, что нам нужно просмотреть какой-то большой файл (пусть это будет /var/log/messages). Введем команду его просмотра:

```
cat /var/log/messages
```

Строки файла пролетят так быстро, что вы не успеете их прочитать. Тогда мы перенаправим вывод этой команды на ввод программы less, которая обеспечивает удобный просмотр файла:

```
cat /var/log/messages | less
```

Суть вы уловили, правда, в этом случае удобнее вызвать less непосредственно и передать ей имя файла:

```
less /var/log/messages
```

Но представим другую ситуацию. Вы настраиваете PPP-соединение и хотите видеть не все строки журнала, а только те, которые относятся к ppp. В этом случае можно перенаправить весь файл на программу grep, которая является текстовым фильтром. Программа grep выделит все строки, содержащие подстроку «ppp». А вот вывод grep уже целесообразно передать программе less для удобного просмотра. Конструкция будет такой:

```
cat /var/log/messages | grep ppp | less
```

Можно вывести вывод команды в файл. Потом этот файл можно передать более опытному коллеге для просмотра или выложить на всеобщее обозрение на форуме. Для вывода в файл используется символ >:

```
cat /var/log/messages | grep ppp > ppp.txt
```

Если файл ppp.txt существовал, его содержимое будет перезаписано. Если вы не хотите перезаписывать содержимое файла, тогда используйте два символа >>:

```
cat /var/log/messages | grep ppp >> ppp.txt
```

6.3.3. Настройка оболочек sh и bash

Как мы уже знаем, bash используется по умолчанию во всех дистрибутивах Linux. Работает bash примерно так же, как и любая другая оболочка. Вы вводите команду, и если она не является внутренней командой bash (вроде cd, pwd и т. д.), то bash пытается найти исполнимый файл программы в каталогах, заданных переменной окружения PATH. В случае успешного поиска bash запускает найденную программу.

В файле /etc/profile содержатся глобальные параметры sh/bash. Эти параметры влияют на всех пользователей системы. В 99% случаев этот файл редактировать вам не придется, а если нужно задать какие-то особые параметры, то лучше это сделать только для вашей учетной записи. Для этого нужно редактировать файл ~/.profile (для sh и bash) или ~/.bash_profile (только для bash). Тильда (~) означает

домашний каталог. Например, вы работаете под учетной записью user, тогда в большинстве случаев домашним каталогом будет /home/user. Вот в нем и нужно искать файлы конфигурации bash для вашей учетной записи.

В листинге 6.3 представлен файл /etc/profile из FreeBSD 8. Комментарии я перевел на русский язык.

Листинг 6.3. Файл /etc/profile

```
# $FreeBSD 8: файл /etc/profile,v 1.14.30.1 2009/08/03 $
#
# Глобальный файл .profile для sh/bash
#
# Специально для фанатов FreeBSD версии 4.2 имеется режим, когда информация о дисках
# показывается в K-блоках. Если вам нравится такое поведение,
# раскомментируйте параметр BLOCKSIZE
#
# BLOCKSIZE=K; export BLOCKSIZE
#
# Разрешить читать системные сообщения
# msgs -f
# Разрешить читать терминальные сообщения
# mesg y
```

Как видите, действительно в этом файле нет ничего интересного. Файл настроек, хранящийся в вашем домашнем каталоге, и то на порядок интереснее. Взгляните на листинг 6.4: в нем представлен файл /root/.profile из моей FreeBSD 8.

Листинг 6.4. Файл /root/.profile

```
# $FreeBSD: src/etc/root/dot.profile,v 1.21.10.1 2009/08/03 08:13:06 kensmith
# Exp $
#
PATH=/sbin:/bin:/usr/sbin:/usr/bin:/usr/games:/usr/local/sbin:/usr/local/bin:~/bin
export PATH
HOME=/root
export HOME
TERM=${TERM:-cons25}
export TERM
PAGER=more
export PAGER
```

Переменная PATH содержит список каталогов, в которых осуществляется поиск запускаемых программ. Когда станете «мегапрограммистом» и напишете хотя бы с десяток bash-скриптов, рекомендую поместить их в отдельный каталог (например, в /usr/share/myscipts) и добавить его в список PATH.

Переменная HOME экспортирует название домашнего каталога. Не рекомендуется изменять это значение, иначе множество скриптов будет работать неправильно. Интересна переменная PAGER, задающая программу для постраничного просмотра. По умолчанию это more, но я бы заменил ее на less:

```
PAGER=less
export PAGER
```

Можно еще изменить стандартный редактор, но об этом мы поговорим в п. 6.5.

Оболочка bash позволяет создавать псевдонимы команд. Псевдонимы команд полезны в двух случаях:

- Когда вам надоело вводить длинные команды — можно создать короткие псевдонимы для каждой длинной команды, и тогда придется вводить всего 1–3 символа.
- Когда вы часто вводите одни и те же параметры при запуске каких-то программ — можно создать псевдоним, состоящий из нужных параметров.

Как правило, псевдонимы определяются в файле .bashrc (для bash) или в .shrc (для sh). Эти файлы вы найдете в своем домашнем каталоге. Если такие файлы отсутствуют, их нужно создать самому:

```
cd ~
touch .bashrc
touch .shrc
```

Нужно отметить, что в Linux часто не существует файл .bash_profile (этот файл читается только оболочкой bash). А если он и существует, то в нем есть одна строка:

```
source ~/.bashrc
```

Как вы можете догадаться, эта строка подразумевает переход на чтение файла .bashrc. Поэтому можно сказать, что вместо файла .bash_profile используется файл .bashrc.

Как правило, в файле .bashrc (или .shrc) описываются псевдонимы команд. Вот несколько полезных псевдонимов, которые вы можете добавить в свой файл конфигурации:

```
alias j=jobs
alias m=$PAGER
alias ll='ls -laFo'
alias l='ls -l'
alias g='egrep -i'
alias grep='grep --color'
alias fgrep='fgrep --color'
alias cp='cp -ip'
alias mv='mv -i'
alias rm='rm -i'
```

Последние три псевдонима сделают работу с системой более безопасной. К каждой команде мы добавили параметр -i, что означает интерактивный режим работы. Команды будут задавать больше вопросов, но зато вы не сможете нечаянно удалить или перезаписать нужные вам файлы[1].

Синтаксис объявления псевдонима прост и понятен:

```
alias <псевдоним>='<команда>'
```

Когда вы вводите псевдоним, bash выполнит указанную команду.

[1] Но по мере появления опыта работы с командной строкой, вы вполне сможете научиться «на автомате» отвечать на вопросы системы и, возможно, когда-нибудь сумеете нечаянно удалить нужный файл, несмотря на все предупреждения. Поэтому будьте внимательны и всегда читайте сообщения и вопросы системы.

Для установки формата приглашения командной строки используется переменная PS1. Обычно формат приглашения такой:

```
имя_пользователя@имя_компьютера:рабочий_каталог
```

Значение PS1 (для такого приглашения) следующее:

```
PS1='\u@\h:\w$'
```

Используйте модификаторы, представленные в табл. 6.4.

Таблица 6.4. Модификаторы приглашения командной строки

Модификатор	Описание
\$	Если UID пользователя равен 0, будет выведен символ #, иначе — символ $
\a	При каждом появлении командной строки компьютер будет издавать писк. В данной таблице приведены далеко не все возможные модификаторы, но этот модификатор здесь не случайно. Если не хотите заработать нервное расстройство, не устанавливайте его. Такой модификатор полезен в одном случае: когда вы запускаете команды, которые долго выполняются, например сборка какой-нибудь программы, ядра, модуля ядра и т. д. Как правило, в этом случае вы вводите команду, а потом ждете минут 10–20, пока она выполнится. Звуковой сигнал отвлечет вас от чашки кофе, и вы снова приступите к работе
\u	Имя пользователя
\H	Полное имя компьютера
\h	Имя компьютера до первой точки
\j	Количество фоновых задач, выполняющихся в оболочке в данное время
\n	Символ новой строки
\r	Возврат каретки
\t	Время в 24-часовом формате (ЧЧ: ММ: СС)
\@	Время в 12-часовом формате (AM/PM)
\l	Имя терминала
\w	Текущий каталог (полный путь)
\W	Текущий каталог (только название каталога, без пути)
\d	Дата в несколько неудобном формате «День недели, Месяц, Число»

После установки переменной PS1 не забудьте ее экспортировать:

```
export PS1
```

История последних команд хранится в файле .bash_history. Если вы не хотите, чтобы кто-нибудь смог просмотреть историю последних команд, удалите или очистите этот файл.

Файл .bash_logout содержит команды, которые обрабатываются при выходе из системы. Поскольку существуют разные способы выхода из системы, то нет никакой гарантии, что указанные команды будут выполнены, поэтому очень часто данный файл вообще отсутствует.

6.3.4. Настройка оболочки tcsh

Как и в случае с bash, у tcsh есть глобальные файлы конфигурации и персональные конфигурационные файлы — в домашнем каталоге каждого пользователя. К глобальным файлам относятся следующие файлы:

- /etc/csh.cshrc — считывается при запуске каждого экземпляра tcsh в интерактивном режиме;
- /etc/csh.login — считывается, только если tcsh является основной оболочкой пользователя (прописана в /etc/passwd для конкретного пользователя);
- /etc/csh.logout — выполняется при выходе из tcsh, при условии, что tcsh является оболочкой по умолчанию для конкретного пользователя.

Если вы хотите, чтобы какая-то команда выполнялась в любом случае (даже если tcsh не является оболочкой по умолчанию) для всех пользователей, тогда вам нужно редактировать первый файл.

Но обычно глобальные файлы конфигурации не изменяются вообще, а изменяются их персональные аналоги в домашнем каталоге пользователя: .cshrc, .login и .logout. Данные файлы являются соответствующими аналогами глобальных файлов, но относятся только к тому пользователю, в домашнем каталоге которого они находятся.

Самое интересное, что порядок считывания глобальных и персональных файлов может отличаться. Логично предположить, что сначала считываются глобальные файлы, а потом — персональные. Но этот порядок может быть изменен при компиляции tcsh. Если в вашей системе tcsh был перекомпилирован, то персональные файлы могут считываться первыми.

В домашнем каталоге также имеется файл .history. В нем хранится история ранее введенных команд. В tcsh есть встроенная команда history, выводящая историю команд, так что просматривать этот файл удобнее именно с помощью команды history. Другие встроенные команды tcsh представлены в табл. 6.5. В таблице 6.5 представлены не все команды, а только те, которые вы будете использовать на практике. Чтобы просмотреть список всех команд, введите команду builtins.

Таблица 6.5. Полезные встроенные команды tcsh

Команда	Описание
alias	Используется для создания псевдонима команды. Команда alias была рассмотрена ранее
alloc	Информация об использовании памяти (практически аналог команды free)
bg	Переводит задачу в фоновый режим
builtins	Выводит список встроенных команд
cd или chdir	Переходит в другой каталог
exit	Выход из tcsh
fg	Делает активной задачу, помещенную ранее в фоновой режим командой bg
hashstat	Отображает статистику механизма хэширования tcsh
history	Показывает ранее введенные команды
jobs	Отображает список задач
kill	«Убивает» процесс, в качестве аргумента нужно передать идентификатор процесса (просмотреть список выполняющихся процессов можно командой ps)
login	Используется для входа под именем другого пользователя. В качестве аргумента указывается имя пользователя

Команда	Описание
logout	Завершает сессию, в большинстве случаев аналог команды exit
nice	Изменяет приоритет процесса. Позже мы поговорим об этой команде подробнее
nohup	Выходит из системы, но не завершает процессы, которые выполняются в фоновом режиме
printenv	Выводит список всех переменных окружения
sched	Простейший планировщик задач. Выполняет команду в указанное вами время, пример: $ sched 19:00 echo "Пора домой!"
set	Объявляет локальную переменную: set color=red
setenv	Объявляет переменную окружения: setenv color red Обратите внимание на огромную разницу в синтаксисе: при определении переменной окружения не нужно использовать знак равенства (=)
stop	Останавливает задачу, которая выполняется в фоновом режиме
time	Секундомер. Запускает программу, переданную в качестве параметра. По завершении работы программы выводит время ее выполнения. Очень полезная команда для программистов и всякого рода тестов производительности (benchmark)
umask	Устанавливает маску прав доступа по умолчанию (см. man umask)
unalias	Удаляет псевдоним команды
unset	Удаляет служебную переменную
unsetenv	Удаляет переменную окружения
where	В качестве параметра нужно передать имя команды. Команда where сообщит, является ли переданная команда встроенной командой или же внешней программой и где находится исполнимый файл этот программы
which	Напоминает обычную утилиту which (см. man which), но работает быстрее, поскольку является встроенной командой. Похожа на команду where

В конфигурационных файлах, как правило, устанавливаются переменные окружения (команда `setenv`), псевдонимы (команда `alias`), служебные переменные (команда `set`). Переменные окружения нужно определять в файле **.login**, который выполняется только один раз — при входе пользователя в систему. Если вы создадите огромный файл **.cshrc** и добавите в него все возможные команды (псевдонимы, переменные окружения и т. д.), то рискуете, что `tcsh` будет работать медленно (хотя на современных компьютерах, думаю, вы не почувствуете разницы).

Формат определения псевдонимов несколько отличается от `bash`:

```
alias <псевдоним> <команда>
```

Символ равенства и кавычки не используются. Вот несколько примеров:

```
alias jjobs -l
alias lals -a
alias lfls -FA
alias llls -lA
alias hhistory 25
```

Переменные окружения тоже определяются иначе. Символ равенства не используется, а команда export тоже не нужна — ее заменяет команда setenv:

```
setenv EDITOR nano
```

Команда set используется для установки служебных переменных, например переменной path (путь поиска программ):

```
set path = (/bin /usr/bin /usr/local/bin /usr/X11R6/bin)
```

6.4. Изменение стандартного редактора во FreeBSD

Многие программы вызывают стандартный редактор vi для изменений той или иной формации. Например, не все конфигурационные файлы рекомендуется редактировать непосредственно, а используются специальные программы, которые блокируют доступ системы к этому файлу (чтобы система не изменила файл, пока вы его редактируете, иначе после сохранения файла может произойти непредвиденная ситуация — содержимое файла предугадать сложно).

Но стандартный редактор vi может понравиться либо настоящим гуру UNIX, либо мазохистам. Более неудобного редактора я не встречал. Это не означает, что я не умею с ним работать, но я предпочитаю делать это как можно реже.

Во FreeBSD вместо vi я предпочитаю использовать более удобный редактор ee, обладающий хоть каким-то интерфейсом пользователя. В Linux я предпочитаю редактор nano, чем-то напоминающий редактор ee.

Чтобы установить редактор по умолчанию, нужно изменить переменную окружения EDITOR. Если у вас tcsh, то откройте файл .cshrc:

```
# ee ~/.cshrc
```

Найдите в этом файле строку:

```
setenv EDITOR vi
```

Данную строку нужно заменить на:

```
setenv EDITOR ee
```

Если такой строки нет, тогда нужно просто добавить строку:

```
setenv EDITOR ee
```

Напомню, что вместо ee можно использовать любой другой редактор, например nano.

При использовании bash нужно открыть файл .bashrc (или .shrc) и добавить в него две строки:

```
EDITOR=nano
export EDITOR
```

Чтобы изменения вступили в силу, выйдите из системы и снова войдите. Настоятельно рекомендую не полениться, а добавить объявление переменной окружения EDITOR в глобальные сценарии для bash и tcsh.

6.5. Русификация консоли во FreeBSD

В современных дистрибутивах Linux практически нет проблем с русификацией. Русифицировано практически все, что возможно, в том числе и консоль. Да, есть мелкие огрехи — иногда разработчики забывают перевести на русский язык то или иное окошко или кнопку в окошке, но это мелочи. Главное, что система, как в графическом интерфейсе, так и в консоли, нормально относится к русскому языку — его можно вводить, он отображается без проблем, и русский текст можно вывести на печать.

А вот во FreeBSD все не так гладко. Консоль нерусифицированна, поэтому сейчас мы исправим эту ситуацию. Русификация консоли FreeBSD — задание вроде бы и не очень сложное, но и тем не менее редактированием одного конфигурационного файла не обойтись. Для лучшего восприятия я разбил весь процесс на несколько этапов.

6.5.1. Этап 1: редактирование файла /etc/rc.conf

Откройте файл rc.conf:

```
# ee /etc/rc.conf
```

В моей системе, если удалить предшествующие комментарии, он выглядел так:

```
hostname="den.localdomain"
ifconfig_em0="DHCP"
keymap="ru.koi8-r"
```

Первые две строчки никак не относятся к русификации, а вот третья строчка задает раскладку клавиатуры. В принципе, можно оставить эту раскладку как есть, но я предпочитаю заменить ее следующей раскладкой:

```
keymap="ru.koi8-r.win"
```

Кодировка символов будет по-прежнему KOI8-R, но расположение русских клавиш будет соответствовать привычной раскладке Windows. Переключение раскладок (русская/английская) будет осуществляться комбинацией Ctrl + Shift. Кстати, если вы поленились указать раскладку ru.koi8-r.win, то переключаться на русский вам придется с помощью клавиши Caps Lock, что не очень удобно.

После этого нужно добавить в rc.conf следующие строки:

```
font8x14="koi8-r-8x14"
font8x16="koi8-r-8x16"
font8x8="koi8-r-8x8"
keyrate="fast"
```

Последний параметр нужен, чтобы клавиатура «работала быстрее».

6.5.2. Этап 2: редактирование файла /etc/passwd

После этого открыть файл /etc/passwd, но его нужно редактировать не обычным редактором, а программой vipw. Нужно для каждого реального пользователя (root и учетные записи всех пользователей, которые вы добавили) установить пятое поле (обычно оно пусто, это так называемое поле Class). Значение этого поля — russian. Введите команду:

```
# vipw
```

В моем случае пришлось изменить всего две записи — root и denis:

```
root:*:0:0:russian:0:0:Charlie &:/root:/bin/csh
denis:*:1000:0:russian:0:0:Denis:/home/denis:/bin/bash
```

Если у вас есть только учетная запись root, то можно ввести вот эту команду:

```
chpass root
```

Когда нужно будет изменить строку `Class`, введите `russian`.

ПРИМЕЧАНИЕ

Программа vipw для редактирования /etc/passwd вызывает стандартный редактор. Если вы в предыдущем разделе не установили редактор ее, будет вызван неудобный редактор vi. Так что настоятельно рекомендую вернуться и установить редактор ее по умолчанию

6.5.3. Этап 3: изменение файла /etc/ttys

Следующий этап — изменение файла /etc/ttys. Откройте этот файл в редакторе `ee`. Этот файл очень большой, поэтому для экономии страниц в книге я его приводить полностью не буду. Вам нужно найти параметры виртуальных консолей ttyv0 — ttyv7 и изменить их локаль. По умолчанию используется локаль `cons25`, а нам нужно изменить ее на `cons25r`. Результат представлен ниже:

```
ttyv0"/usr/libexec/getty Pc"cons25ron secure
# Virtual terminals
ttyv1"/usr/libexec/getty Pc"cons25ron secure
ttyv2"/usr/libexec/getty Pc"cons25ron secure
ttyv3"/usr/libexec/getty Pc"cons25ron secure
ttyv4"/usr/libexec/getty Pc"cons25ron secure
ttyv5"/usr/libexec/getty Pc"cons25ron secure
ttyv6"/usr/libexec/getty Pc"cons25ron secure
ttyv7"/usr/libexec/getty Pc"cons25ron secure
```

ПРИМЕЧАНИЕ

Хотя это и не имеет никакого отношения к локализации, вам наверное интересно, что означает слово «secure» в контексте консоли. Слово «secure» с английского языка переводится как «безопасный». Выходит, что консоль, напротив которой указано «secure», является безопасной, а, значит, на ней можно регистрироваться как root. Иногда в целях безопасности имеет смысл ограничить число консолей, позволяющих входить в систему как root.

6.5.4. Этап 4: редактирование файлов конфигурации оболочек

Теперь нужно отредактировать файлы /etc/profile (если вы используете `bash`) и /etc/csh.cshrc (если у вас `tcsh`). В /etc/profile нужно добавить строки:

```
export LANG=ru_RU.KOI8-R
export LC_ALL=ru_RU.KOI8-R
```

В /etc/csh.cshrc нужно добавить вот эти строки:

```
setenv LANG ru_RU.KOI8-R
export LC_ALL ru_RU.KOI8-R
```

В файле ~/.profile нужно изменить значение переменной TERM:

```
TERM=${TERM:-cons25r}
```

6.5.5. Этап 5: настройка клавиш клавиатуры

Осталось совсем немного, но ведь нам не нужны половинчатые решения? Отредактируем файл /etc/inputrc для правильной реакции системы на нажатие файла Del. В этот файл нужно добавить следующие строки:

```
$if term=cons25r
"\C-?": delete-char
$endif
```

Если файл /etc/inputrc не существует, его нужно прежде создать:

```
# touch /etc/inputrc
```

6.5.6. Этап 6: перезагрузка

Можно перезагрузить сервис syscons

```
# /etc/rc.d/syscons restart
```

А можно последовать примеру Microsoft и перезагрузить всю систему:

```
# reboot
```

В результате вся консоль (в том числе вывод команд и даже редактор ee!) будет русифицирована. На рис. 6.1 представлен русифицированный редактор ee, а на рис. 6.2 — вывод команды ls (тоже на русском языке):

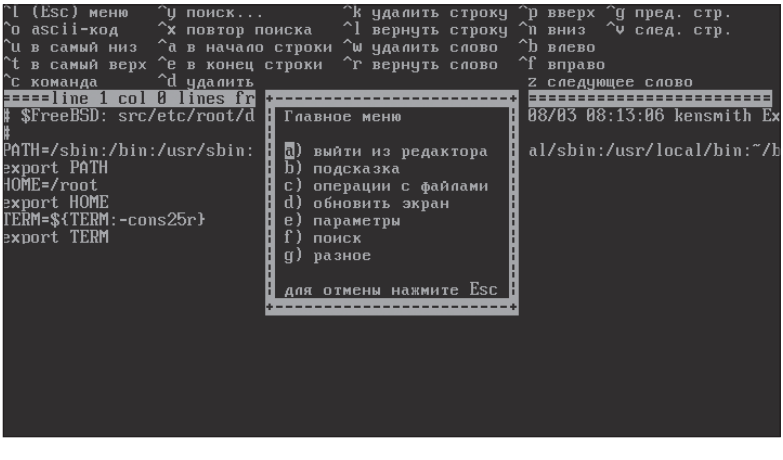

Рис. 6.1. Редактор ee русифицирован

```
den# ls -l
total 6
-rw-------  1 root  wheel     59 10 ноя 11:28 .bash_history
-rw-r--r--  2 root  wheel    794 10 ноя 11:27 .cshrc
-rw-------  1 root  wheel      0 10 ноя 11:31 .history
-rw-r--r--  1 root  wheel    151 17 сен 19:53 .k5login
-rw-r--r--  1 root  wheel    299 17 сен 19:53 .login
-rw-r--r--  2 root  wheel    262 10 ноя 11:20 .profile
den#
```

Рис. 6.2. Вывод команды ls на русском языке

Вот теперь наша консоль полностью на русском языке.

6.6. Полезные команды

Настало время рассмотреть полезные команды, которые пригодятся вам каждый день. В этой главе не будут представлены команды для работы с файлами и каталогами — им посвящена вся следующая глава. Вообще, будьте готовы каждый день узнавать все новые и новые команды UNIX, так как их намного больше, чем вы можете себе представить.

6.6.1. Общие команды

Команда clear: очистка консоли

Бывшим пользователям MS-DOS знакома команда `cls`, очищающая экран. В UNIX для этой цели используется команда `clear`:

```
$ clear
```

Команда date: установка и просмотр даты

Команда `date` служит для установки даты и времени (для этого нужно обладать правами root) и для отображения текущей даты и времени (просмотреть дату могут все пользователи). Пример использования команды:

```
$ date
Tue Oct 5 11:12:23 EEST 2010
```

Установка даты и времени осуществляется в формате:

```
# date ccyymmddHHMMss
```

Здесь:

- cc — столетие, но поскольку обычно не нужно устанавливать дату в таких глобальных масштабах, столетие можно не указывать;
- yy, mm, dd — год, месяц и день. Все числа нужно указывать с предшествуемым нулем, если нужно. Например, для установки даты 1 января 2011 года нужно указать 110103, а не 1113;
- HH, MM, ss — часы, минуты, секунды. Как и в случае с датой, не забывайте о предшествующих нулях.

Следующая команда установит дату 1 января 2011 года и время 02:03:

```
# date 110103020300
```

Если вам нужно указать только дату, время можно не указывать, например:

```
# date 110105
```

Команды df и free: просмотр ресурсов

Команда `df` выводит информацию об использовании дискового пространства каждой смонтированной файловой системы, а команда `free` — информацию об использовании оперативной памяти. Вывод обеих команд интуитивно ясен, поэтому подробно их рассматривать не будем, во всяком случае, сейчас. А вот если вам ничего не ясно, тогда внимательно прочитайте главу 7, в которой мы поговорим об устройстве файловой системы. После ее прочтения вы получите ответы на возникшие вопросы.

Команда echo: вывод сообщений

Обычно команда `echo` используется в сценариях для вывода на консоль небольших сообщений. Но ее удобно использовать для перенаправления некого текста на ввод другой команды. Представим, что у нас есть команда, которая после запуска долго что-то делает (скажем, минут 5–10), потом задает вопрос, на который нужно ответить `Y` (да) или `N` (нет), а после еще что-то долго делает. Общее время выполнения этой команды может составить около 20 минут. А ведь это время можно использовать для заслуженного перерыва и отойти подальше от компьютера. Но ждать первые 5–10 минут не хочется — ведь нужно только нажать `Y`. Вот для этого и можно использовать команду `echo` примерно так:

```
$ echo y | команда
```

Символ `y` будет передан на ввод указанной команды так, если бы вы его нажали лично.

Команды exit и logout: выход из оболочки и системы

Команда `exit` обеспечивает выход из оболочки. Если используемая оболочка является оболочкой по умолчанию (так называемой login-shell), то будет выполнен выход из системы.

Бывает так, что у вас оболочка по умолчанию одна (`tcsh`), но с ее помощью вы запустили `bash`. Тогда команда `exit` выполнит выход из `bash`, и вы вернетесь в `tcsh`. Чтобы выйти из системы, нужно еще раз ввести `exit`. Поэтому проще использовать команду `logout` для немедленного выхода из системы.

Команда md5: проверка контрольной суммы

Представим, что вы закачали ISO-образ какого-нибудь дистрибутива Linux или новой версии FreeBSD. Как проверить, правильно ли он был закачан. Вроде бы и размер совпадает, но перед установкой нужно знать точно, что образ закачан без ошибок — это сэкономит вам время в дальнейшем. Иначе возможна ситуация, когда вы запишите образ на болванку, начнете установку, а на 97-м проценте инсталляции программа установки сообщит, что не может прочитать какие-то файлы с диска.

Обычно разработчики сообщают контрольную сумму ISO-образа, вычисленную с помощью MD5. Далее все просто. Вы скачиваете образ к себе на компьютер, вводите следующую команду:

```
$ md5 <имя_файла>
```

Сравниваете ее вывод с контрольной суммой, представленной на сайте. Если они совпадают, значит, файл скачан без ошибок.

Команда passwd: изменение пароля

Команда `passwd` может использоваться как для изменения пароля текущего пользователя, так и для изменения пароля любого другого пользователя, но для этого нужно обладать правами root:

```
$ passwd
# passwd den
```

Первая команда изменяет пароль текущего пользователя (вы изменяете свой собственный пароль). Вторая команда изменяет пароль пользователя den, но обратите внимание на приглашение командной строки: команда должна быть выполнена от имени root.

6.6.2. Команды управления процессами: ps, kill, killall, nice, top

В UNIX у каждого процесса (запущенной программы) есть свой уникальный номер — идентификатор процесса (PID, Process ID). Зная идентификатор процесса, мы можем его завершить. Конечно, такое принудительное завершение необходимо, если мы не можем завершить процесс обычным способом (например, когда последний завис).

Узнать номер процесса можно с помощью команды `ps`. Для вывода списка всех процессов используется параметр `-e`:

```
ps -e
```

Команда `ps -e` выводит PID-процесса, имя терминала, на котором запущен процесс (не выводится для графических приложений), занимаемое процессорное время и команду, которой был запущен процесс (рис. 6.3).

Для завершения процесса используется команда `kill`:

```
kill <PID>
```

Процесс может запускать другие процессы — процессы-потомки. Для завершения процесса и всех его потомков используется параметр `-9` команды `kill`:

```
kill -9 <PID>
```

Кроме команды `kill` есть более удобная команда — `killall`, позволяющая «убить» процесс не по номеру, а по имени. Например, завис у вас файловый менеджер `mc`. Вы вместо того чтобы вычислять его номер, просто введите команду:

```
killall mc
```

Напомню, что вы можете «убить» только те процессы, которые вы запустили. Если вы не являетесь владельцем процесса, а процесс все же нужно завершить, тогда вам понадобятся полномочия пользователя root, получить которые можно с помощью команды `sudo`:

```
sudo kill -9 <PID>
```

```
den@den-desktop:~$ ps -e
  PID TTY          TIME CMD
    1 ?        00:00:01 init
    2 ?        00:00:00 migration/0
    3 ?        00:00:00 ksoftirqd/0
    4 ?        00:00:00 watchdog/0
    5 ?        00:00:00 events/0
    6 ?        00:00:00 khelper
    7 ?        00:00:00 kthread
   30 ?        00:00:00 kblockd/0
   31 ?        00:00:00 kacpid
   32 ?        00:00:00 kacpi_notify
  106 ?        00:00:00 kseriod
  129 ?        00:00:00 pdflush
  130 ?        00:00:00 pdflush
  131 ?        00:00:00 kswapd0
  132 ?        00:00:00 aio/0
 1937 ?        00:00:00 ksuspend_usbd
 1938 ?        00:00:00 khubd
 2148 ?        00:00:00 ata/0
 2149 ?        00:00:00 ata_aux
 2320 ?        00:00:00 kjournald
 2519 ?        00:00:00 udevd
 3495 ?        00:00:00 kpsmoused
 3567 ?        00:00:00 kgameportd
 3841 ?        00:00:00 kjournald
 3846 ?        00:00:00 kjournald
 3848 ?        00:00:00 kjournald
 4157 tty4     00:00:00 getty
 4158 tty5     00:00:00 getty
 4161 tty2     00:00:00 getty
 4162 tty3     00:00:00 getty
 4163 tty1     00:00:00 getty
 4164 tty6     00:00:00 getty
 4410 ?        00:00:00 acpid
 4547 ?        00:00:00 dbus-daemon
```

Рис. 6.3. Список процессов

Для завершения работы графических программ намного удобнее использовать программу `xkill`. Для ее запуска нажмите Alt + F2, в появившемся окне введите `xkill` и нажмите Enter. Указатель мыши станет похожим на черный череп. Им нужно щелкнуть по окну программы, которую вы хотите принудительно завершить. Если вы передумали, то просто нажмите правую кнопку мыши.

С помощью команды `nice` можно установить приоритет выполнения процесса. Например, вы запустили перекодировщик видео на ночь и хотите предоставить ему все ресурсы, чтобы до утра видео точно было перекодировано. Или же, наоборот, вы запустили перекодировщик утром и хотите установить минимальный приоритет, чтобы он не мешал вам работать. Максимальный приоритет равен –20 (отрицательное значение). Минимальный — +19 (положительное). Чтобы установить отрицательный приоритет, вам нужны права root, даже если вы являетесь владельцем процесса, например:

```
sudo nice -n -10 codec
nice -n 19 codec
```

Для вывода подробной информации о запущенных процессах используется команда `top` (рис. 6.4). Данная программа выводит:

- Время работы системы (с момента запуска) — time.
- Количество зарегистрированных пользователей (имеется в виду не количество учетных записей, а количество пользователей, которые в данный момент работают с системой) — users.

- Общую загрузку системы — load average.
- Информацию о задачах — всего (total), количество выполняющихся процессов (running), спящих процессов (sleeping), остановленных (stopped), количество процессов-зомби (zombie). Процессом-зомби называется ситуация, когда процесс уже завершил свою работу, но информация о нем еще осталась в таблице процессов. Процессы-зомби — нормальное явление в Linux.
- Информацию об использовании оперативной памяти (Mem) — всего (total), использовано (used), свободно (free).
- Информацию об использовании разделов подкачки (Swap).
- Информацию о процессах: PID, имя владельца (USER), приоритет (PR и NI), использованное процессорное время (%CPU), использованная память (%MEM), команда запуска процесса. Назначение остальных колонок вывода top не очень интересно — вы всегда сможете прочитать о них в man top.

```
top - 05:51:52 up 27 min,  3 users,  load average: 0.26, 0.17, 0.11
Tasks: 101 total,   2 running,  98 sleeping,   0 stopped,   1 zombie
Cpu(s):  1.0%us,  0.0%sy,  0.0%ni, 99.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:    775684k total,   529960k used,   245724k free,    15348k buffers
Swap:   787072k total,        0k used,   787072k free,   354220k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 4750 root      15   0  168m  17m 7968 S  0.7  2.3   0:37.51 Xorg
 4797 cupsys    20   0  4680 1984 1444 S  0.3  0.3   0:00.71 cupsd
 6330 den       15   0  2316 1152  880 R  0.3  0.1   0:00.02 top
    1 root      18   0  2908 1844  524 S  0.0  0.2   0:01.33 init
    2 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/0
    3 root      34  19     0    0    0 S  0.0  0.0   0:00.00 ksoftirqd/0
    4 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 watchdog/0
    5 root      10  -5     0    0    0 S  0.0  0.0   0:00.06 events/0
    6 root      10  -5     0    0    0 S  0.0  0.0   0:00.02 khelper
    7 root      14  -5     0    0    0 S  0.0  0.0   0:00.00 kthread
   30 root      10  -5     0    0    0 S  0.0  0.0   0:00.00 kblockd/0
   31 root      20  -5     0    0    0 S  0.0  0.0   0:00.00 kacpid
   32 root      20  -5     0    0    0 S  0.0  0.0   0:00.00 kacpi_notify
  106 root      10  -5     0    0    0 S  0.0  0.0   0:00.00 kseriod
  129 root      18   0     0    0    0 S  0.0  0.0   0:00.00 pdflush
  130 root      15   0     0    0    0 S  0.0  0.0   0:00.00 pdflush
  131 root      13  -5     0    0    0 S  0.0  0.0   0:00.00 kswapd0
```

Рис. 6.4. Информация о процессах

Для каждого процесса выводится два показателя приоритета — PR и NI, что немного сбивает с толку. PR — это приоритет процесса. Показатель NI означает, насколько больше или меньше процесс «нравится» (от слова nice) системе и другим процессам, то есть насколько приоритет процесса был изменен командой nice. Значение 0 (значение NI по умолчанию) в колонке NI означает, что приоритет процесса не изменялся с момента запуска. Если значение NI положительное, значит, процесс может «уступить место» другим процессам, независимо от своих потребностей в процессорном времени. Такой процесс (с положительным NI) «нравится» другим процессам — он ведь уступил место в переполненном автобусе. А вот процесс с отрицательным NI будет выполняться быстрее, но будет меньше нравиться другим процессам — он похож на человека, который вне очереди получил те или иные услуги — он получил их первым, но от этого менее нравится всем остальным, кто честно ждал своей очереди.

По параметру wa (время простоя, в строке CPU) можно косвенно судить о возможном скором выходе из строя жесткого диска. Если параметр wa часто превы-

шает 80%, то процессор много времени проводит в ожидании ввода/вывода. А это может быть в том случае, если жесткий диск уже начал отказывать. На всякий случай проверьте поверхность жесткого диска программой badblocks и сделайте резервную копию всех данных.

Вам интересно, почему вы работаете один, а программа сообщает, что количество зарегистрированных пользователей — 2 или 3? Введите команду w и вы узнаете, какие пользователи (и сколько раз) зарегистрированы в системе (рис. 6.5).

В моем случае количество зарегистрированных пользователей равно 3. Все 3 раза зарегистрировался я сам — один раз в графическом интерфейсе и запустил два терминала (запуск терминала равносилен регистрации в системе).

```
den@den-desktop:~$ w
 06:07:10 up 42 min,  3 users,  load average: 0,50, 0,20, 0,12
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
den      :0       -                05:25   ?xdm?   1:38m  0.31s x-session-manag
den      pts/0    :0.0             05:26   20:34m  0.25s  0.01s man nice
den      pts/1    :0.0             05:33    0.00s  0.21s  0.00s w
den@den-desktop:~$
```

Рис. 6.5. Информация о работающих в системе пользователях

6.6.3. Статистика: uptime, users, w, whoami, who

Команда uptime отображает время работы системы, общее число зарегистрированных в системе пользователей и общую нагрузку за последние 1, 5 и 15 минут. Вот пример вывода этой программы:

```
2:20PM up 5:19, 1 user, load averages: 0.02, 0.03, 0.01
```

Проанализируем вывод программы: текущее время 14:20, время работы системы 5 часов и 19 минут, в системе зарегистрирован всего один пользователь. Последние три числа — это средняя загрузка системы за 1, 5 и 15 минут. Показатели очень низкие, поэтому вам не о чем беспокоиться.

```
Хотите узнать, кто сейчас зарегистрирован в системе? Тогда введите команду users:
# users
root
```

Как видно из вывода users, в системе зарегистрирован только пользователь root — о нем и сообщила команда uptime (1 user).

Более подробную информацию о зарегистрированных пользователях сообщает ранее рассмотренная команда w. Преимущество этой команды очевидно: она не только сообщает имена зарегистрированных пользователей, но и способы их регистрации. Рассмотрим еще один небольшой пример вывода этой команды:

```
# w
2:24PM up 5:23, 2 users, load averages: 0.00, 0.00, 0.00
USER TTY FROM LOGIN@ IDLE WHAT
root v0 – 02:10PM – w
den v1 – 02:23PM – mc
```

Команда w выводит имена зарегистрированных пользователей, виртуальную консоль (TTY), адрес удаленного узла (FROM), время регистрации, время простоя

и текущий процесс, запущенный на виртуальной консоли. Если в поле FROM значение :0.0 или подобное — это означает, что пользователь работает в графическом режиме.

А теперь посмотрите на рис. 6.5. На нем видно, что команда w была выполнена в Linux: это понятно по именам терминалов. А сейчас мы выполнили эту команду во FreeBSD. Назначение столбцов такое же, как и в случае с Linux. Вот только в Linux выводится информация об использовании процессорного времени. В странице руководства man о полях JCPU и PCPU сказано только «CPU times» без объяснения, что означает тот или иной столбец. Давайте заполним этот пробел:

- JCPU — это время, которое использовано всеми процессами, закрепленными за tty. Данное время не включает фоновые задания, которые выполняются в данный момент времени.
- PCPU — время, использованное текущим процессом, указанным в поле WHAT.

Команда who также отображает список зарегистрированных в системе пользователей, но ее вывод гораздо скромнее: имя пользователя, консоль и время регистрации. Пример вывода программы:

```
# who
root ttyv0 Oct 3 14:10
den ttyv1 Oct 3 14:23
```

Еще одна полезная команда — whoami. Она сообщает имя текущего пользователя. Неужели она помогает имеющим проблемы с памятью? Вы вошли в систему и не знаете, под каким пользователем[1]. На самом деле эта команда чаще используется в сценариях оболочки, чтобы можно было проверить, кто запустил сценарий.

6.6.4. Команды для работы с текстом

В UNIX очень много команд для работы с текстом. Сейчас мы рассмотрим необходимый любому администратору минимум, но если вы хотите стать настоящим UNIX-гуру, настоятельно рекомендую ознакомиться с программированием на языке awk (gawk) — это язык обработки текстовых файлов. С его помощью можно легко организовать обработку файлов журналов системы.

Команда grep: фильтр

С командой grep мы уже сталкивались. Команда grep принимает со стандартного ввода текст и выводит строки, содержащие подстроку, переданную grep в качестве аргумента. Вот несколько полезных примеров использования grep:

```
ps ax | grep pppd
dmesg | grep usb
cat /var/log/messages | grep usb
```

[1] Что вполне возможно и при хорошей памяти — например, если вы входите одновременно на нескольких виртуальных консолях под совершенно разными именами пользователей для тестирования какой-нибудь программы, а в приглашении командной строки имен пользователей нет, только знак доллара.

В первом случае вывод команды ps ax (список всех запущенных в системе процессов) перенаправляется программе grep, на консоль будет выведена информация о демоне pppd, если он запущен.

Во втором случае выводится список сообщений ядра, относящийся к usb. Данную команду полезно вводить после подключения USB-устройства, чтобы понять, подключилось ли устройство. Можно также ввести третью команду. Но в этом случае получите сведения не о текущем подключенном устройстве, а обо всех USB-устройствах, которые подключались к системе с момента последней очистки /var/log/messages.

Команды tail и head: начало и конец файла

Команда head выводит первые 10 строк текстового файла, а команда tail — последние 10 строк. Количество строк регулируется параметром -n. Обычно администраторы используют команду tail для просмотра больших файлов журналов, чтобы увидеть последние записанные в журнал строки. Вот пример вывода последних 15 строк файла /var/log/messages:

```
tail -n 15 /var/log/messages
```

Данную команду полезно вводить сразу после какого-нибудь выполненного действия, например после запуска сетевой службы, установки соединения с Интернетом и т. д. Вы увидите, были ли какие-то предупреждения при запуске службы и какой IP-адрес присвоен вашему компьютеру при установке соединения с провайдером.

Команда wc: подсчет слов в текстовом файле

Вы написали статью в журнал? Иногда размер материала оценивается по количеству символов (с пробелами или без), а иногда — по количеству слов. Команда wc поможет вам подсчитать количество слов в файле. Файл должен быть текстовым, для других форматов используйте средства редактора этого формата (редакторы OpenOffice.Org и MS Word имеют средства для подсчета количества символов, слов, строк, страниц и т. д.). Пример использования:

```
$ wc article_about_unix.txt
```

Команда less: удобный просмоторщик текстовых файлов

Команда more (появилась значительно раньше less) используется для постраничного просмотра файла. Но ее недостаток в том, что просмотр именно постраничный, да еще и можно листать файл только вперед, а вот если вы просмотрели что-то на предыдущей странице, придется выходить из программы и запускать ее заново, что очень неудобно. На всякий случай: клавиша Пробел листает страницы вперед, а Q используется для выхода из программы.

Программа less намного удобнее. Она позволяет листать файл не только вперед, но и назад. Стрелками ↑ и ↓ можно листать файл построчно, а Пробел используется, как и в программе more, для перехода к следующей странице.

Вот несколько примеров использования программы:

```
$ dmesg | less
$ less article.txt
```

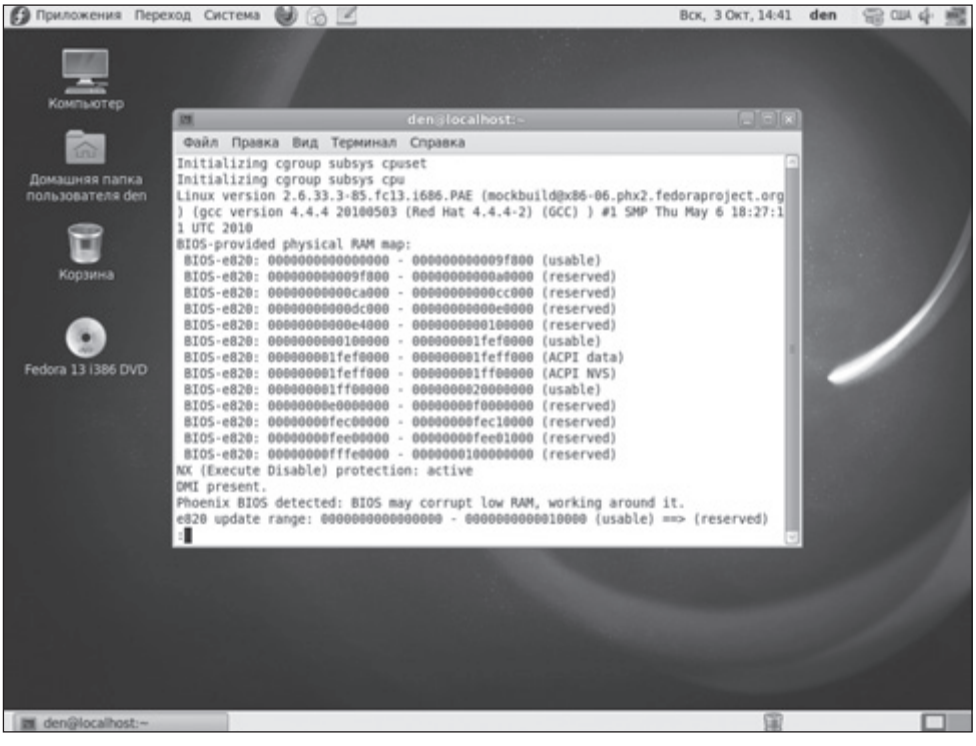

Рис. 6.6. Команда dmesg | less в терминале

Первая команда перенаправляет вывод команды `dmesg` (вывод довольно большой, см. рис. 6.6) на стандартный ввод программы `less`, что позволяет удобно просматривать сообщения ядра. А вторая команда используется непосредственно для просмотра файла `article.txt`

Команда diff: сравнение файлов

Иногда очень полезно сравнить две версии файла. Для сравнения файлов используется команда `diff`. Формат вызова команды:

```
diff [опции] файл1 файл2
```

Результат программы — отчет о сравнении файлов. Отличающиеся строки будут отмечены символами **>** и **<**. Строка, найденная в первом файле, отмечается символом **<**, а строка из второго файла — символом **>**.

Опция `-b` игнорирует пробельные символы, к которым относят символ пробела и табуляции. Опция `-B` игнорирует пустые строки. У программы `diff` есть много разных опций, со всеми ними вы можете ознакомиться в справочном руководстве (`man diff`). Конкретный выбор опций зависит от желаемого результата и формата файлов.

II

Администрирование UNIX

Файловая система UNIX

7

7.1. Типы файловых систем

7.1.1. Родные файловые системы Linux

С самого своего создания родной файловой системой для Linux была файловая система ext. Позже ее место заняла файловая система ext2. Эта файловая система долгое время являлась самой удачной файловой системой для Linux. Даже когда появилась следующая версия — ext3, — ext2 оставалась в лидерах. Пользователи и администраторы не спешили переходить на ext3, поскольку ext2 была все еще быстрее. Снижение производительности в случае новой файловой системы ext3 произошло из-за введения журнала операций — файловая система стала более надежной, но и более медленной. Позже проблема с производительностью была исправлена в ext4, но самое интересное, что старая ext2, разработанная еще в 1993 году, уступает современной файловой системе ext4 только в тестах на чтение. Так что когда будете устанавливать Linux в следующий раз, выбирайте или ext2 или ext4, ext3 — самый худший вариант.

Вот некоторые характеристики файловых систем ext2:

- Максимальный размер тома — 2–32 Тбайт.
- Максимальный размер файла 16 Гбайт — 2 Тбайт.
- Максимальное количество файлов — 10^{18}.
- Максимальная длина имени файла — 255 байт.

Файловая система ext4 гораздо современнее:

- Максимальный размер тома — 1 эксабайт или 1024 петабайта или 1 048 576 Тбайт (таких жестких дисков еще не создано).
- Максимальный размер файла (при использовании размера блока в 4 Кбайт) — 16 Тбайт (10^{12} байт).
- Максимальное количество файлов — 4 млрд.

Не думаю, что вы превысите эти ограничения[1]. Помимо файловых систем ext2–ext4 в качестве корневой файловой системы в Linux могут использоваться ReiserFS, XFS и JFS. Две последние файловые системы — это разработки Silicon Graphics и IBM соответственно — в Linux они не прижились. Хотя при желании их можно использовать. А вот файловая система Reiserfs стала популярной благодаря возможности хранить в одном блоке данные, относящиеся к разным файлам, что позволяет более рационально использовать место на диске.

Сейчас поясню, что имеется в виду. Весь жесткий диск на физическом уровне разбивается на секторы — блоки по 512 байт. Файловая система разбивает дисковое пространство на логические блоки, которые не могут быть меньше одного физического блока. Зато они могут быть больше; как правило, логический блок больше физического на степень двойки — в 2, 4, 8 раз. Чем больше размер логического блока, тем быстрее будет работать файловая система — ведь за одну и ту же операцию копирования блока будет скопировано, скажем, 1024 или 2048 байт, а не 512. Зато чем меньше размер блока, тем эффективнее используется дисковое пространство. Допустим, вам нужно записать файл размером всего 12 байт. Но он займет весь логический блок. Если логический блок равен 512 байт, то «коту под хвост» уйдет 500 байт, а если размер блока 1024 байт, то 1012 байт будет потрачено впустую. Именно поэтому многие администраторы и просто опытные пользователи Linux стараются при установке Linux создать три раздела: один для корневой файловой системы, второй для подкачки и третий для пользовательских данных. Для первого раздела (где есть обычно множество мелких файлов) имеет смысл установить размер блока 512 байт, а для третьего, где могут быть файлы большого размера — ISO-образы, фильмы, музыкальные файлы, нужно установить размер блока 1024–2048 байт — так будет быстрее. Размер блока указывается при создании файловой системы (при форматировании раздела).

Файловая система ReiserFS (кстати, когда говорят «ReiserFS», то имеют в виду третью версию файловой системы Reiser, а четвертая называется Reiser4, но сейчас нам версия Reiser не важна) снимает это ограничение и позволяет в один логический блок упаковать несколько мелких файлов. За это ее и любят пользователи. К тому же она тоже является журналируемой, поэтому можете использовать ReiserFS без потери надежности. Что же касается производительности, то она на уровне ext3.

Прежде чем перейти к следующему разделу, развею миф об отсутствии фрагментации ReiserFS. То, что для этой файловой системы не существует дефрагментатора, не означает, что нет самой фрагментации. А лучший способ фрагментации ReiserFS — это полный дамп раздела и его последующее восстановление. Не очень удобно, зато после этого ReiserFS начинает работать быстрее.

Какую файловую систему выбрать? Несмотря на все преимущества ReiserFS, лично мой выбор — ext4, — пока это лучшая файловая система для Linux.

[1] Но не исключено, что вы все-таки столкнетесь с ситуацией, когда и такие параметры будут считаться недостаточными. Например, за счет объединения нескольких физических дисков в один логический с помощью специальных контроллеров или программ ниже уровня файловой системы. Тогда могут потребоваться тома и больших размеров. Впрочем, и приведенные данные тоже лучше уточнить применительно к своему дистрибутиву.

7.1.2. Родные файловые системы BSD

Для FreeBSD вы можете выбрать одну из трех файловой систем в качестве корневой: UFS (UNIX Filesystem), UFS2 и ZFS. Первая безнадежно устарела, вторая является 64-разрядной версией первой, поэтому в большинстве случаев является оптимальной, поскольку поддерживает разделы большого размера.

На старом оборудовании UFS/UFS2 работают медленно. Представьте себе маленькую черепашку. UFS2 проигрывает даже ext2, но другого выбора у вас нет, поскольку на старом «железе» ZFS еще хуже. А вот если у вас современный жесткий диск SATA/SATA-II, то UFS2 будет работать гораздо шустрее. Поэтому на современном компьютере вы можете выбрать либо UFS, либо ZFS. Если производительность UFS2 вас не будет устраивать, можно перевести ее в асинхронный режим ввода/вывода (как это сделать, будет показано дальше), но помните, что в этом режиме есть риск потери данных, поэтому асинхронный режим нужно использовать на свой страх и риск. Могу сказать точно: в асинхронном режиме ваша файловая система будет работать быстрее — проверено на личном опыте. А вот насчет надежности... Самое страшное, что может произойти, — это отключение питания. Поэтому прикупите хороший ИБП желательно с функцией информирования об отключении питания, чтобы ваш сервер смог корректно завершить работу.

Файловая система ZFS разрабатывалась с нуля, в отличие от UFS, которая была разработана еще в 80-х годах прошлого века, а UFS2 — всего лишь количественное, но не качественное обновление UFS. А ZFS изначально разрабатывалась с учетом новых аппаратных средств. Но тут и вся загвоздка: эта файловая система очень требовательна к оперативной памяти. Если у вас ОЗУ меньше 1 Гбайт, используйте UFS2, иначе вы вместо ожидаемого прироста производительности все равно получите черепашку.

7.1.3. Прочие файловые системы. Включение поддержки прочих файловых систем

Помимо родных файловых систем Linux и FreeBSD поддерживают другие популярные в компьютерном мире файловые системы: CD9660 (CD/DVD-диски), VFAT/FAT32 (Windows-разделы и флешки), NTFS (Windows-разделы) и т. д. Поддерживаемых файловых систем много, но мы остановимся только на тех, которые нас больше всего интересуют.

В Linux все очень и очень просто. Практически все сразу встроено в ядро (а то, что не встроено, загружается автоматически), поэтому для включения поддержки той или иной файловой системы вам ничего не нужно делать. Более того, монтирование (то есть подключение файловой системы к корневой файловой системе Linux) осуществляется автоматически. Все происходит как в Windows: вы подключили флешку — откроется окошко с содержимым флешки, вставили DVD-диск — появится аналогичное окошко, подключили еще один винчестер — на рабочем столе увидите пиктограмму доступа к нему. Только не забывайте перед извлечением носителя щелкать правой кнопкой мыши на значке файловой системы и выбирать команду `Извлечь` или `Отключить` — в зависимости от дистрибутива

и типа носителя. Поэтому далее все, что касается монтирования, будет в первую очередь относиться к FreeBSD, а не к Linux.

Во FreeBSD не все так просто. Если при попытке монтирования файловой системы (раздел 7.4) вы получите сообщение об ошибке, тогда вам нужно включить поддержку нужной вам файловой системы. Включить поддержку — это загрузить тот или иной модуль ядра. Чтобы включить ту или иную файловую систему, нужно в /boot/loader.conf добавить следующие строки:

```
# Включаем подержку FAT/FAT32 (Windows-разделы и флешки)
msdosfs="YES"
# Включаем поддержку Ext2/3/4
ext2fs_load="YES"
# Поддержка ReiserFS
reiserfs_load="YES"
# Поддержка ZFS
zfs_load="YES"
```

После редактирования файла /boot/loader.conf нужно перезагрузить компьютер:

```
# reboot
```

О поддержке NTFS мы еще поговорим, поскольку она заслуживает отдельного разговора.

7.2. Устройство файловой системы UNIX

7.2.1. Отличие организации дискового пространства Linux от BSD

В главе 2 вы уже познакомились с отличием организации дискового пространства Linux и FreeBSD. Давайте освежим полученные знания. В Windows мы привыкли разбивать жесткий диск на логические диски. Купив жесткий диск на 120 Гбайт, вы непременно разбиваете его на 4 логических диска по 30 Гбайт каждый, полученные логические диски в Windows получают имена C:, D:, E: и F:. Логические диски также называются разделами (partitions).

В Linux схема примерно такая же: когда вы устанавливаете Linux, то создаете как минимум два раздела — один для корневой файловой системы, второй для подкачки.

Решение не очень экономное, поскольку, фактически, эти два раздела принадлежат одной операционной системе, но ведь занято целых два раздела жесткого диска! Все мы знаем, что на одном жестком диске можно создать четыре первичных раздела и один расширенный[1]. В расширенном можно создавать другие разделы. Но если вы по незнанию заняли все четыре первичных раздела, то уже

[1] При использовании таблицы разделов типа MBR можно создать от одного до четырех разделов одинаковых или разных типов, из которых только один может быть «расширенным» разделом для систем MS-DOS (обязательно второй) и Windows.

не сможете установить на компьютер еще одну операционную систему, которая может загружаться только с первичного раздела.

FreeBSD предлагает иное решение. То, что мы привыкли называть разделами, в FreeBSD называется слайсами. Все, что нужно FreeBSD, помещается в одном слайсе. А уже внутри себя слайс делится на BSD-разделы — их может быть до 8 штук. Экономно? И я так считаю: ведь установи вы две FreeBSD на один компьютер, было бы занято 16 разделов, а так — всего два.

Как мы уже узнали из главы 2, жесткий диск, подключенный к первому IDE-контроллеру как Primary Master, во FreeBSD называется /dev/ad0. Предположим, что на нем установлены Linux и FreeBSD. Linux занимает два раздела (слайса) — /dev/ad0s1 и /dev/ad0s2, а FreeBSD — всего один — /dev/ad0s3. Внутри BSD-слайса может быть определено до восьми разделов — ad0s3a—ad0s3h. Раздел ad0s3b — это подкачка (имя b зарезервировано для подкачки). А вот какой из разделов, /dev/ad0s1 или /dev/ad0s2, используется в качестве подкачки в Linux, сказать сложно. В Linux нет никаких требований к имени раздела, раздел с корневой файловой системой может у вас называться ad0s1, а раздел с подкачкой — ad0s7. Все имена, приведенные здесь, соответствуют именам во FreeBSD. В Linux они называются иначе, а как именно — вы узнаете чуть позже.

7.2.2. Требования к именам файлам

Максимальная длина имени файла в UNIX — 254 символа (для некоторых файловых систем — 256 символов). Это максимальная длина полного пути к файлу. Например, рассмотрим имя /root/file.txt. Длина имени файла — 14 символов (/root/file.txt), а не 8 символов (file.txt). Поэтому 254 символа при большой вложенности каталогов и длинных именах этих самых каталогов не очень-то и много.

Имя файла может содержать любые символы, кроме / \ ? < > * " |. Вы также можете использовать кириллицу (при желании), но с учетом переноса файлов на другие файловые системы лучше воздержаться от названий файлов на русском языке[1].

Разделение элементов пути производится с использованием слэша (/), а не обратного слэша, как в Windows (\).

Имена файлов в UNIX чувствительны к регистру: Unix.txt, UNIX.TXT, uNiX.tXt — это три разных файла. И еще: в отличие от Windows в UNIX нет понятия «расширение файлов». В Windows расширением (типом) файла считаются последние несколько символов после последней точки. А в UNIX все точки (если они вообще есть) относятся к имени файла. В UNIX можно выделить только полное имя файла (содержит полный путь к файлу) или сокращенное имя (только название файла, без пути к нему).

[1] Следует заметить, что фактические ограничения как на длины собственно имен файлов и полных имен (с путями), так и на используемые символы — это сочетание требований собственно файловой системы, ее реализации в конкретной операционной системе (ограничения, накладываемые версией ядра и утилит), а также конкретным командным интерпретатором (например, используется ли в нем символ ? для указания на «любой допустимый символ»).

7.2.3. Иерархия файловой системы UNIX

Первым делом нужно поговорить о корневой файловой системе Linux. Пусть вы установили Linux на какой-то раздел жесткого диска. В этом разделе будет создана корневая файловая система. Корневая файловая система обозначается так: /.

Чтобы использовать другую файловую систему (не корневую), например файловую систему вашего Windows-раздела, ее нужно *подмонтировать* к корневой файловой системе. При загрузке ядро Linux монтирует корневую файловую систему, далее вы можете подмонтировать другие файловые системы к корневой файловой системе. Файловая система монтируется к определенному каталогу, который называется *точкой монтирования*. Точкой монтирования может быть любой каталог. Главное, чтобы он существовал на момент монтирования. Например, если вы подмонтировали свой Windows-раздел к каталогу /mnt/windows, то потом через этот каталог вы сможете обратиться к файлам и каталогам, находящимся на Windows-разделе. Подробно о монтировании мы поговорим чуть позже, а пока рассмотрим содержимое корневого каталога UNIX.

В корневом каталоге обязательно будут следующие каталоги:

- /bin — в этом каталоге вы найдете стандартные программы UNIX (cp, ls и т. д.).
- /boot — содержит файлы загрузчика UNIX.
- /dev — здесь вы найдете файлы устройств (см. ниже).
- /etc — каталог конфигурационных файлов.
- /home — каталог пользовательских файлов. При создании пользователя в этом каталоге создается подкаталог /home/<имя_пользователя> — это и есть домашний каталог созданного пользователя. Во FreeBSD каталог /home — это только ссылка на каталог /usr/home.
- initrd — данный каталог используется при загрузке системы, не удаляйте его! Во FreeBSD этого каталога нет. В некоторых дистрибутивах Linux он есть, а в некоторых файлы, хранящиеся в этом каталоге, перемещены в каталог /boot. В любом случае, если этот каталог в вашей системе есть, удалять его не нужно.
- /lib — содержит библиотеки и модули ядра.
- /lost+found — здесь вы найдете файлы, которые система восстановила после некорректного размонтирования файловой системы (опять выключили свет?). Во FreeBSD этого каталога нет.
- /media — используется средствами автоматического монтирования для монтирования сменных носителей, хотя все зависит от настроек системы — в вашей системе сменные носители могут монтироваться к другим каталогам.
- /mnt — данный каталог традиционно содержит точки монтирования — каталоги, к которым монтируются файловые системы.
- /opt — сюда устанавливаются дополнительные программные пакеты. В Linux этот каталог используется редко, а вот во FreeBSD, возможно, найдутся такие программы, которые будут установлены в каталог /opt.

- /proc — псевдофайловая система, предоставляющая информацию о процессах.
- /root — домашний каталог администратора — пользователя root.
- /sbin — содержит системные утилиты. Программы из этого каталога имеет право запускать только root.
- /sys — каталог псевдофайловой системы sysfs, предоставляющей информацию об устройствах и драйверах.
- /tmp — «мусорка», то есть каталог для временных файлов (не путать с корзиной!).
- /usr — пользовательские программы, документация, исходные коды ядра и программ и т. д.
- /var — в этом каталоге содержатся постоянно изменяющиеся данные, например очередь печати, почтовые ящики, журналы системы и т. д.

7.2.4. Имена файлов устройств дисковых накопителей

Linux

В предыдущем разделе было отмечено, что в каталоге /dev/ находятся файлы устройств. Да, все правильно: в Linux каждое устройство (даже процессор) имеет свой файл. Не верите? Загляните в каталог /dev/cpu/0 — вы найдете в нем информацию о вашем первом процессоре!

Через файл устройства можно или влиять на работу устройства или просто получать информацию о работе устройства.

Имена файлов жестких дисков ATA (IDE) и IDE-приводов CD/DVD начинаются с символов hd (от hard disk). После hd следует буква, которая зависит от того, как подключено устройство:

- hda — Primary Master — «Первичный мастер» — устройство, подключенное как мастер к первому IDE-контроллеру;
- hdb — Primary Slave — «Первичный подчиненный» — устройство, подключенное как подчиненное к первому IDE-контроллеру;
- hdc — Secondary Master — «Вторичный мастер» — устройство, подключенное как мастер ко второму IDE-контроллеру;
- hdd — Secondrary Slave — «Вторичный подчиненный» — устройство, подключенное как подчиненное ко второму IDE-контроллеру.

Цифра после имени устройства — это имя раздела, например hda1 — это первый раздел (обычно это диск C:) на первом IDE-устройстве.

Со SCSI-дисками (SATA-дисками) все аналогично, только первые две буквы не hd, а sd. Буква после sd задает порядок подключения, а цифра — номер раздела. Например, /dev/sda1.

Казалось бы, все просто. Но потом в голову разработчиков Linux пришла идея разработать менеджер устройств udev. И тут началась настоящая путаница. Дело в том, что после принятия udev все жесткие диски, вне зависимости от типа кон-

троллера (PATA, SATA, SCSI, флешка), стали называться /dev/sdX, где X — буква. Если в вашей системе есть винчестер (заметьте, я не уточняю его интерфейс), то он будет называться /dev/sda. Первый раздел на этом винчестере (куда обычно устанавливается Windows) будет называться /dev/sda1. Когда вы подключите флешку, то ей будет присвоено имя /dev/sdb.

А что делать, если в системе установлено несколько жестких дисков? Как понять, какому жесткому диску какое было присвоено имя? Понять это можно, только проанализировав вывод dmesg, например:

```
# dmesg | grep sda
# dmesg | grep sdb
```

Мы получаем информацию сначала о диске sda, потом о диске sdb. При выводе информации будет выведено название производителя, модель жесткого диска, емкость диска — по этим параметрам можно определить, «кто есть кто».

Вот еще некоторые имена устройств, которые вам необходимо знать:

- /dev/cdrom — ссылка на устройство привода CD/DVD. Если вам нужно обратиться к CD/DVD, а вы не знаете, как подключен привод, можете использовать ссылку /dev/cdrom;
- /dev/fd0 — первый дисковод для дискет (в Windows — это диск A:);
- /dev/fd1 — первый дисковод для дискет (в Windows — это диск B:);
- /dev/ttyS0 — первый COM-порт (COM1);
- /dev/ttyS1 — второй COM-порт (COM2).
- /dev/modem — ссылка на устройство модема, например на /dev/ttyS1, если модем подключен к COM2

FreeBSD

В отличие от Linux во FreeBSD все намного проще. SCSI-винчестеры называются /dev/daN, где N — это номер, который зависит от подключения винчестера к контроллеру. Флешки (USB-накопители) приравниваются к SCSI-устройствам, поэтому если у вас есть SCSI-диск и вы подключили флешку, то SCSI-диску будет присвоено имя /dev/da0, а флешке — /dev/da1. Если же SCSI-диска у вас нет, то флешке будет присвоено имя /dev/da0.

IDE-диски называются /dev/adN:

- /dev/ad0 — Primary Master;
- /dev/ad1 — Primary Slave;
- /dev/ad2 — Secondary Master;
- /dev/ad3 — Secondary Slave.

SATA-диски приравняли к PATA (IDE). Жесткий диск, подключенный к первому SATA-контроллеру, называется /dev/ad4. Поскольку к IDE-контроллеру можно подключить два диска, а к SATA — только один, то имя /dev/ad5 для аналогии с IDE пропускается. Поэтому диск, подключенный ко второму SATA-контроллеру, получит название /dev/ad6. Аналогично, имя /dev/ad7 не используется.

CD/DVD-приводы, подключенные к SATA/PATA-контроллеру, называются /dev/acdN, где N — это число. А привод, подключенный к SCSI-контроллеру, будет называться /dev/cdN, например /dev/cd0.

7.2.5. Домашний каталог пользователя

Как мы уже знаем, в каталоге /home хранятся домашние каталоги пользователей. Название подкаталогов в этом каталоге соответствует имени пользователя. Например, каталог /home/den принадлежит пользователю den. Только пользователь den (и понятно, root) имеет полный доступ к этому каталогу.

В домашнем каталоге пользователя хранятся настройки программ, которые он использует (например, личные настройки среды KDE, OpenOffice и т. д.), а также личные файлы пользователя. В этом каталоге много скрытых файлов и каталогов, как правило, это конфигурационные файлы разных программ, содержащие личные настройки пользователя.

Если вы помните, в Windows был атрибут файла «Скрытый». Если атрибут установлен, то файл по умолчанию не был виден при просмотре содержимого каталога (пока вы не включали режим отображения скрытых файлов). В Linux такого атрибута нет, но скрыть файл подобным образом все же можно. Если имя файла/каталога начинается с точки, то файл считается скрытым и его по умолчанию не показывают файловые менеджеры.

Домашний каталог в Linux обозначается так: ~. Например, если пользователь den введет в консоли команду `cd ~`, то он перейдет в каталог /home/den.

7.3. Работаем с файлами и каталогами

UNIX первоначально был ориентирован именно на работу в командной строке. Даже когда появился графический интерфейс X Window (позже его переименовали в X.Org), большинство операций выполнялось в командной строке. В современных дистрибутивах все операции с файлами и каталогами можно выполнить в графическом режиме, но во FreeBSD (на сервере) обычно графический интерфейс не устанавливается, поэтому с файлами и каталогами придется работать в командной строке.

7.3.1. Команды для работы с файлами

Для работы с файлами вы можете использовать следующие команды:

- `touch <имя_файла>` — создание пустого файла;
- `cat <имя_файла>` — вывод содержимого текстового файла;
- `tac <имя_файла>` — вывод содержимого текстового файла в обратном порядке, то есть сначала выводится последняя строка файла, затем предпоследняя и т. д.;
- `more <имя_файла>` — постраничный просмотр текстового файла, переход к следующей странице осуществляется нажатием клавиши Пробел;
- `less <имя_файла>` — удобная утилита просмотра текстовых файлов. Для перемещения по файлу используйте клавиши PageUp и PageDown;

- `cp <файл1> <файл2>` — копирование файла `<файл1>` в файл `<файл2>`. В именах файлов можно использовать маски, например `cp *.doc /media/disk`;
- `mv <файл1> <файл2>` — перемещение/переименование файла `<файл1>` в файл `<файл2>`;
- `rm <имя_файла>` — удаление файла;
- `which <программа>` — определение каталога, в котором находится файл на диске;
- `ln <имя_файл> <имя_ссылки>` — создание ссылки на файл/каталог. По ссылке можно обращаться к файлу. Грубо говоря, ссылка похожа на ярлык в Windows;
- `mcedit <имя_файла>` — удобный текстовый редактор. Входит в состав пакета `mc`.

ПРИМЕЧАНИЕ

Если вы хотите подробнее узнать о том, что делает та или иная команда, обратитесь к справочной системе Linux: `man <имя_команды>`. Справка на русском, поэтому проблем с переводом не будет — было бы желание читать справку[1].

Особого внимания заслуживает команда `ln`. Она используется для создания ссылок на файл или каталог. Ссылки в Linux бывают двух типов: жесткие и мягкие (их еще называют символическими). Жесткая ссылка — это просто еще одно имя файла/каталога. То есть у одного файла/каталога может быть несколько имен. Это очень удобно, к тому же защищает файл от случайного удаления: файл нельзя удалить, пока на него ссылается хотя бы одна жесткая ссылка. Тут все просто: поскольку есть хотя бы одно из имен файла, то его удалить нельзя. Как только последнее имя будет удалено, то файл будет удален с диска.

Символическая («мягкая») ссылка представляет собой отдельный файл на диске. В этом файле содержится имя файла или каталога, на который она ссылается. Преимущество символической ссылки в том, что она может ссылаться на файлы из другой файловой системы, а жесткая ссылка — нет. Зато символическая ссылка никак не защищает файл от удаления. Она существует отдельно от файла — как отдельный, независимый файл. Вы можете удалить файл, а на него будут ссылаться сотни ссылок. Понятно, что после удаления файла они будут ссылаться в никуда.

По умолчанию команда `ln` создает жесткую ссылку. Если нужно создать символическую ссылку, то следует указать параметр `-s`:

```
ln -s файл ссылка
```

Для создания жесткой ссылки на каталог (а не файл) нужно использовать параметр `-d`:

```
ln -d каталог ссылка
```

[1] К сожалению, далеко не во всех дистрибутивах и не для всех команд справка переведена на русский язык, и далеко не всегда вариант на русском языке устанавливается по умолчанию.

7.3.2. Команды для работы с каталогами

Теперь поговорим о командах для работы с каталогами:

- `mkdir <каталог>` — создание каталога;
- `ls <каталог>` — вывод оглавления каталога;
- `cd <каталог>` — переход в каталог (смена каталога);
- `rmdir <каталог>` — удаление *пустого* каталога;
- `rm -r <каталог>` — рекурсивное удаление непустого каталога (будут удалены все файлы и все подкаталоги).

7.3.3. Понятие владельца файла. Права доступа к файлам и каталогам

В UNIX есть понятие владельца и прав доступа. Владелец — это пользователь, создавший файл или каталог. Владелец вправе назначить права доступа к файлу, то есть определить, кто будет иметь доступ к файлу, а кто — нет.

Если вы являетесь владельцем файла, то можете его «подарить» другому пользователю. Это можно сделать с помощью команды `chown` (сокращение от change owner):

```
chown пользователь файл
```

Например:

```
chown user report.txt
```

После выполнения этой команды новым владельцем файла report.txt станет пользователь user.

Теперь поговорим о правах доступа к файлам и каталогам. Предположим, что у нас есть файл report.txt. С помощью команды `ls -l report.txt` можно узнать его права доступа. Вывод команды будет примерно такой:

```
-rw-r----- 1 den den 2007-05-27 05:26 report.txt
```

Строка «`-r--r-----`» обозначает набор прав доступа. Первый символ (`-`) обозначает, что перед нами обычный файл, а не каталог (в случае каталога на этом месте стоял бы символ d). Остальные символы можно разделить на три группы:

- `rw-` — владелец имеет право читать (r — read) и изменять (w — write) файл;
- `r--` — члены группы владельца могут только читать файл;
- `---` — остальные пользователи не имеют к файлу никакого доступа (при обращении к файлу они увидят сообщение о том, что доступ запрещен).

Кроме прав на чтение и изменение файла есть право на его выполнение (оно обозначается символом `x`, например `rwx`). Право на выполнение устанавливается для программ и сценариев командной оболочки. Если файл не является программой или сценарием, понятно, запустить его не получится. Как и в случае, если он является программой или сценарием, но право на выполнение не установлено.

Для каталога право на выполнение разрешает переходить в данный каталог и выводить его оглавление.

Разберемся, как устанавливаются права доступа. Рассмотрим отдельный набор прав доступа, например rw-. Мысленно заменим все символы «-» на «0», а все остальные — на «1». На месте права чтения стоит символ «r», а не «-», поэтому представляем, что на его месте 1. Далее опять 1. Потом 0. Выходит число 110. Если вы догадались, это число в двоичной системе, но переводить его нужно не в десятичную, а в восьмеричную. Если вы прогуляли курс информатики в школе (кажется, там рассматривали системы счисления), тогда я вам помогу:

- 000 — 0;
- 001 — 1;
- 010 — 2;
- 011 — 3;
- 100 — 4;
- 101 — 5;
- 110 — 6;
- 111 — 7.

А теперь установим права доступа к файлу file.txt:

```
chmod 666 file.txt
```

В данном случае используется набор прав 110 (цифра 6 в десятичной системе), что разрешает чтение и запись файла владельцу (первая 6), группе владельца (вторая 6) и всем остальным пользователям (третья 6). Нужно отметить, что режим 666 не является безопасным. По возможности старайтесь его не использовать. Оптимальным для обычных файлов является режим 640 — чтение и запись разрешены владельцу (6); группе пользователей, к которой принадлежит владелец, разрешено только чтение (4), а остальным пользователям запрещен доступ к файлу (0).

Режим 777 для файлов означает не только полный доступ (чтение и запись) всем желающим, но и выполнение файла всем желающим. Тоже довольно небезопасный режим. Но 777 обычно используется не для файлов, а для каталогов — бит «выполнения» для каталога (третья 1 в режиме 111) означает право просмотра содержимого каталога.

Обычный пользователь может изменить права доступа (либо «подарить» файл с помощью команды chown) только своих файлов и каталогов. Пользователь root может изменить права доступа и владельца любого файла в системе.

7.3.4. Файловый менеджер mc

Облегчить работу с файлами и каталогами в консоли призван файловый менеджер Midnight Commander (команда mc), который как две капли воды напоминает старого знакомого Norton Commander. Установите пакет mc с помощью вашего менеджера пакетов. Во FreeBSD для установки mc можно использовать команду:

```
# pkg_add -r mc
```

Файловый менеджер Midnight Commander изображен на рис. 7.1.

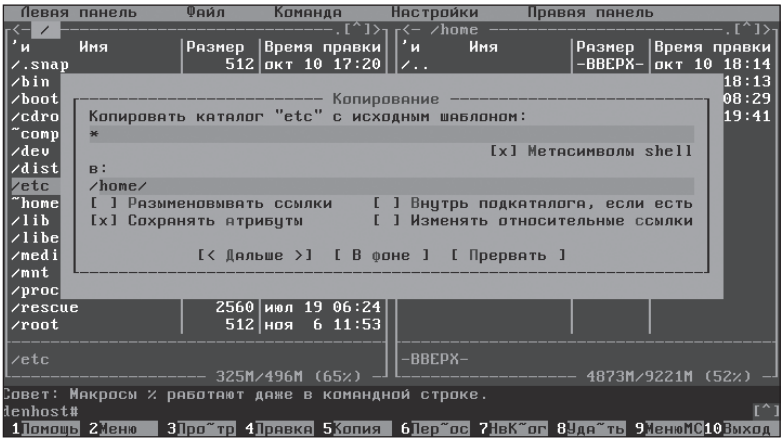

Рис. 7.1. Файловый менеджер Midnight Commander

7.4. Команда mount: монтирование файловых систем

7.4.1. Команды mount и umount

Монтирование — это очень важная операция в мире Unix/Linux. Чтобы получить доступ к другой файловой системе, например к Windows-разделу, к CD/DVD, к файловой системе Flash-диска (флешки), нужно эту файловую систему подмонтировать к корневой файловой системе. Каталог, к которому подмонтирована файловая система, называется точкой монтирования. Через этот каталог будет производиться обращение к файлам и каталогам подмонтированной файловой системы.

Сейчас все объясню на примере. Пусть у нас есть Flash-диск, на котором имеется файл film.avi. В Linux Flash-диску соответствует файл устройства /dev/sda (если нет других SATA/SCSI-дисков). На Flash-диске обычно всего один раздел, поэтому нужное нам устройство будет называться /dev/sda1. Файловую систему этого устройства мы подмонтируем к каталогу /mnt/flash. Важно, чтобы этот каталог существовал на момент монтирования, поэтому его нужно заранее создать.

Рассмотрим следующие команды:

```
# mkdir /mnt/flash
# mount /dev/sda1 /mnt/flash
# cp /mnt/flash/film.avi ~
```

Сначала мы создаем точку монтирования — каталог /mnt/flash, затем монтируем устройство /dev/sda1 (так будет называться флешка в Linux, но имя устройства у вас может быть другим). Далее мы копируем файл film.avi из точки монтирования в домашний каталог (в данном случае в домашний каталог пользователя root).

Если же в системе установлен один SATA-диск, флешка будет называться /dev/sdb1 (точнее первый и часто единственный раздел на флешке), поэтому для ее монтирования нужно ввести команду:

```
# mount /dev/sdb1 /mnt/flash
```

ПРИМЕЧАНИЕ

В большинстве случаев в Linux осуществляется автоматическое монтирование как сменных, так и стационарных носителей. Но не во всех дистрибутивах все проходит гладко. Так, в Debian в графическом режиме без проблем монтируются флешки, отформатированные в NTFS. А вот стационарные носители — разделы жесткого диска, которые должны монтироваться «в один клик», не монтируются, а вместо этого пользователь видит сообщение об ошибке. Выход из ситуации есть. На ваш выбор предоставляю вам три варианта. Первый — вручную монтировать NTFS-раздел. Подойдет, только если вы очень редко используете этот раздел. Второй — прописать этот раздел в /etc/fstab для монтирования при загрузке системы (это мой выбор). Третий — настроить систему так, чтобы она умела монтировать стационарные носители так же, как и сменные. Сей процесс описан в статье http://gettoknowlinux.blogspot.com/2009/10/ntfs.html

В FreeBSD команда монтирования будет выглядеть немного иначе:

```
# mount -t msdosfs /dev/da0s1 /mnt/flash
```

Желательно указать тип файловой системы, имя устройства во FreeBSD тоже будет другим — /dev/da0s1. Во FreeBSD вы можете использовать следующие типы файловых систем:

- msdosfs — для файловой системы FAT/FAT32;
- ext2fs — для файловых систем ext2/3/4;
- reiserfs — для ReiserFS;
- cd9660 — для CD9660 (CD/DVD);
- ufs — используется по умолчанию, можно не указывать.

В общем случае формат вызова mount выглядит так:

```
# mount [опции] устройство точка_монтирования
```

Размонтировать файловую систему можно командой umount. В качестве параметра этой команде нужно передать или смонтированное устройство или точку монтирования. В нашем случае команда размонтирования будет выглядеть так:

```
# umount /mnt/flash
```

На момент размонтирования устройство не должно использоваться, иначе размонтировать его не получится.

Во FreeBSD при подключении флешки на консоль root ядро выведет сообщения, описывающие подключенный носитель. Не пугайтесь — все нормально. Сообщения будут примерно такими:

```
da0 at umass-sim0 bus 0 scbus1 target 0 lun 0
da0: < silicon-power PMAP> Removable Direct Access SCSI-0 device
da0: 40.000MB/s transfers
da0: 3822MB (7827456 512 byte sectors: 255H 63S/T 487C)
GEOM: da0: partition 1 does not end on a track boundary.
```

7.4.2. Файл /etc/fstab

В файле /etc/fstab содержится список автоматически монтируемых файловых систем. Все файловые системы, перечисленные в этом файле, монтируются при загрузке операционной системы. В моей системе файл /etc/fstab выглядит так:

```
/dev/sda6 / ext4 relatime 1 1
/dev/cdrom /media/cdrom auto umask=0022,users,iocharset=utf8,noauto,ro,exec 0 0
/dev/sda1 /mnt/win_c vfat umask=0022,iocharset=utf8 0 0
/dev/sda8 /mnt/win_d vfat umask=0022,iocharset=utf8 0 0
/dev/sda9 /mnt/win_e vfat umask=0022,iocharset=utf8 0 0
/dev/sda10 /mnt/win_f vfat umask=0022,iocharset=utf8 0 0
/dev/sda11 /mnt/win_g vfat umask=0022,iocharset=utf8 0 0
/dev/sda12 /mnt/win_h vfat umask=0022,iocharset=utf8 0 0
none /proc proc defaults 0 0
/dev/hda7 swap swap defaults 0 0
```

Формат этого файла очень прост:

```
устройство точка_монтирования тип_файловой_системы опции дамп проверка
```

Из этого файла следует, что

- корневая файловая система расположена на устройстве /dev/sda6;
- CD/DVD-диск будет монтироваться к каталогу /media/cdrom;
- Windows-разделы C, D, E, F, G и H будут монтироваться к соответствующим каталогам /mnt/win_?, где ? — буква раздела;
- псевдофайловая система proc монтируется к каталогу /proc;
- раздел /dev/sda7 используется как раздел подкачки.

Последние два поля (дамп и проверка) определяют, будет ли файловая система включена в дамп (при выполнении команды dump) и будет ли она проверена при загрузке командой fsck. Для корневой файловой системы эти два поля должны быть установлены в 1 и 1. Для всех других файловых систем, для которых нужно включить дамп и проверку, — в 2 и 2. Если дамп и проверка не нужны, нужно установить значения 0 и 0.

Вот самые распространенные опции монтирования файловых систем:

- ro — монтирование в режиме «только чтение»;
- rw — режим «чтение/запись», используется по умолчанию, можно не указывать;
- iocharset — кодировка символов (для правильного отображения символов имен файлов);
- umask — маска создания файлов, можно не указывать;
- noauto — файловая система не должна монтироваться автоматически при загрузке системы (ни при вводе команды mount -a), используется для сменных носителей;
- async — асинхронный режим ввода/вывода, в BSD (для UFS/UFS2) позволяет повысить производительность файловой системы;
- noatime — не устанавливать время последнего доступа к файлу, позволяет тоже немного повысить производительность файловой системы.

Как вы уже успели заметить, пример файла /etc/fstab был приведен из Linux. Это видно по именам устройств и типу корневой файловой системы (ext4). Во FreeBSD файл fstab имеет такой же формат, но будет отличаться другими именами устройств и другими типами файловой системы. Например, если вам нужно смонтировать файловую систему FAT32 во FreeBSD, то вместо vfat вы должны указать тип msdosfs. Вот пример моего файла /etc/fstab из FreeBSD:

```
# Device      Mountpoint   FStype     Options     Dump  Pass#
/dev/ad0s1b   none         swap       sw          0     0
/dev/ad0s1a   /            ufs        rw          1     1
/dev/ad0s1e   /tmp         ufs        rw          2     2
/dev/ad0s1f   /usr         ufs        rw          2     2
/dev/ad0s1d   /var         ufs        rw          2     2
/dev/acd0     /cdrom       cd9660     ro,noauto   0     0
/dev/da0s1    /mnt/usb     msdosfs    noauto      0     0
```

Как видите, файл отличается только именами устройств и другими файловыми системами. Обратите внимание на последние две строки. Для монтирования DVD и флешки при наличии этих записей можно воспользоваться командами:

```
# mount /cdrom
# mount /mnt/usb
```

Другими словами, мы можем не указывать ни тип файловой системы, ни имя устройства, если они указаны в /etc/fstab. Такой трюк можно использовать и в Linux, но, как правило, в этом нет смысла, поскольку сменные носители в Linux монтируются автоматически.

ПРИМЕЧАНИЕ

Просмотреть, сколько осталось свободного места на разделах, можно командой df -h. Параметр -h необходим, чтобы команда df выводила информацию в виде, удобном для человека, то есть в мегабайтах, а не в блоках.

7.4.3. Монтирование NTFS-раздела

Монтирование NTFS-раздела всегда было каверзным вопросом. В Linux и во FreeBSD есть стандартный драйвер ntfs, предназначенный для монтирования NTFS-разделов, но этот драйвер поддерживает только чтение с раздела. А если смонтировать файловую систему в режиме rw и попытаться записать что-то на диск, последствия такого действия будут непредсказуемыми.

Однако можно использовать драйвер ntfs-3g, позволяющий производить корректную запись на NTFS-раздел. Сначала разберемся, как использовать этот драйвер в Linux. В большинстве случаев пакет ntfs-3g установлен по умолчанию, если это не так, его нужно установить. Далее для монтирования NTFS-раздела введите команду:

```
sudo mount -t ntfs-3g <имя_устройства> /mnt/windows
```

Для автоматического монтирования раздела при загрузке в /etc/fstab нужно добавить строки:

```
# /dev/sda1 NTFS
/dev/sda1 /mnt/windows ntfs-3g defaults,nls=utf8,umask=007,gid=46 0 0
```

Теперь разберемся с установкой драйвера ntfs-3g в FreeBSD. Сначала нужно установить порт fusefs-ntfs:

```
# cd /usr/ports/sysutils/fusefs-ntfs/
# make install clean
```

Следующий шаг — обеспечить загрузку демона fusefs, добавив вот эту строку в /etc/rc.conf:

```
fusefs_enable="YES"
```

Теперь запустим сам демон:

```
# /usr/local/etc/rc.d/fusefs start
```

Осталось только подмонтировать NTFS-раздел, мы используем опции rw (чтение/запись) и locale (устанавливаем верную кодировку символов):

```
# ntfs-3g -o rw,locale=ru_RU.UTF-8 имя_устройства /mnt
```

7.4.4. Монтирование ISO-образа

В UNIX можно подмонтировать ISO-образ к корневой файловой системе и просмотреть его содержимое. Конечно, изменить содержимое образа не получится, но зато вы хоть сможете просмотреть, что внутри, и при необходимости скопировать из ISO-образа файлы и каталоги.

Во FreeBSD для монтирования образа введите команды:

```
# mdconfig -a -t vnode -u0 -f имя_образа.iso
# mount -t cd9660 /dev/md0 /mnt
```

После этого обратиться к содержимому образа можно через каталог /mnt. Для размонтирования образа нужно ввести команды:

```
# umount /dev/md0
# mdconfig -d -u0
```

В Linux смонтировать ISO-образ можно командой:

```
# mount -o loop -t iso9660 имя_образа.iso /mnt
```

Обратиться к файлам образа можно через каталог /mnt. Размонтировать образ можно командой:

```
# umount /mnt
```

7.5. Разметка и проверка жесткого диска

Для проверки файловой системы UFS нужно использовать программу fsck:

```
# fsck <имя_устройства>
```

Устройство должно быть размонтировано на момент проверки. Для проверки устройства, содержащего корневую файловую систему, вам нужно загрузиться с LiveCD и инициировать проверку устройства.

Проверить другие файловые системы можно командами:

```
# fsck_msdosfs <имя_устройства>; FAT/FAT32
# reiserfsck <имя_устройства>; ReiserFS
# fsck.ext2 <имя_устройства>; ext2
# fsck.ext3 <имя_устройства>; ext3
# fsck.ext4 <имя_устройства>; ext4
```

Последние три команды во FreeBSD не установлены по умолчанию, для их установки нужно установить пакет e2fsprogs:

```
# pkg_add -r e2fsprogs
```

Однако я настоятельно рекомендую проверять «чужие» файловые системы в их родной операционной системе — так меньше вероятность того, что при проверке что-то пойдет не так.

Теперь поговорим о добавлении нового жесткого диска. В Linux, как и в Windows, все достаточно просто. Алгоритм будет примерно таким:

- Выключить питание компьютера.
- Подключить винчестер.
- Включить компьютер, загрузить Linux.
- Если винчестер уже разбит на разделы, то на рабочем столе вы увидите значки для доступа к этим разделам. Возможно, вам захочется исправить точки монтирования в файле /etc/fstab.
- Если винчестер не разбит на разделы, то нужно запустить программу fdisk <имя_устройства> и создать разделы. Хотя, учитывая, что программа fdisk неудобна, вы будете использовать программы gparted (в Debian/Ubuntu/Denix), diskdrake (в Mandriva), YaST (в OpenSUSE), Disk Druid (Fedora, CentOS), partitionmorpher (в других дистрибутивах). Все эти программы довольно просты в использовании, поскольку они обладают графическим интерфейсом. Не все эти программы хороши, например partitionmorpher откровенно слабовата, Disk Druid не умеет изменять размер разделов, но в любом случае они лучше, чем текстовый fdisk. Самая лучшая программа разметки — в Mandriva, второе место — gparted, но она пока не умеет работать с LVM (хорошо, что LVM на рабочих станциях пока редкость). В крайнем случае, можно использовать загрузочный диск с PartitionMagic — кроме создания разделов еще и отформатируете их.
- После создания разделов нужно создать файловую систему. Здесь можно использовать программу mkfs (см. man mkfs) для создания родной файловой системы Linux.
- После редактирования таблицы разделов вам нужно модифицировать /etc/fstab, добавив точки монтирования.

Во FreeBSD все не так просто. После физического подключения жесткого диска он никак не влияет на работу системы — как будто его нет. Если на жестком диске есть разделы, то вам нужно модифицировать /etc/fstab для того, чтобы «прописать» в нем новый жесткий диск. Например, вы подключили второй SATA-винчестер (/dev/ad6), на котором есть один слайс (s1), на котором есть два раздела (d и e). Для их подключения нужно добавить в /etc/fstab строки:

```
/dev/ad6s1d/mnt/ad6s1dufsrw22
/dev/ad6s1e/mnt/ad6s1eufsrw22
```

Сохраните файл и создайте точки монтирования:

```
# mkdir /mnt/ad6s1d
# mkdir /mnt/ad6s1e
```

Теперь для монтирования новых разделов введите команду:

```
# mount -a
```

Конечно, можете по примеру Microsoft перезагрузить компьютер, но это явно лишнее действие.

А вот если разделов нет, тогда придется их создать. Проще всего (чтобы не использовать других программ) взять стандартный конфигуратор `sysinstall`. Запустите его и выберите команду Configure (рис. 7.2). Далее нужно выбрать команду Fdisk (рис. 7.3).

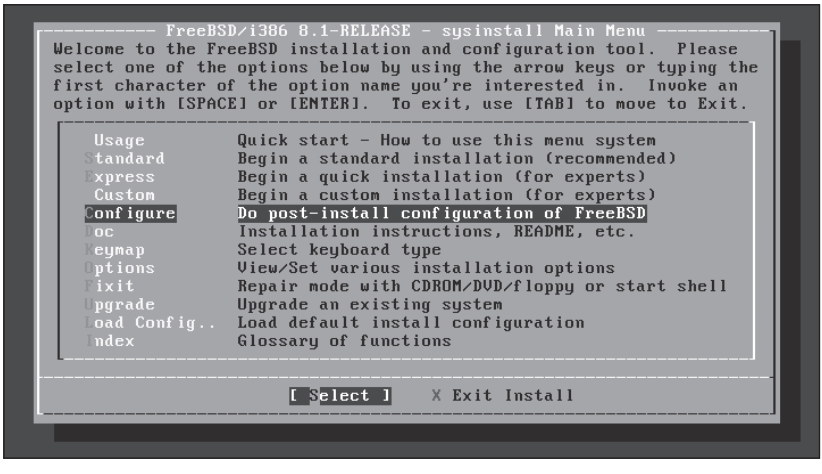

Рис. 7.2. Выберите команду Configure

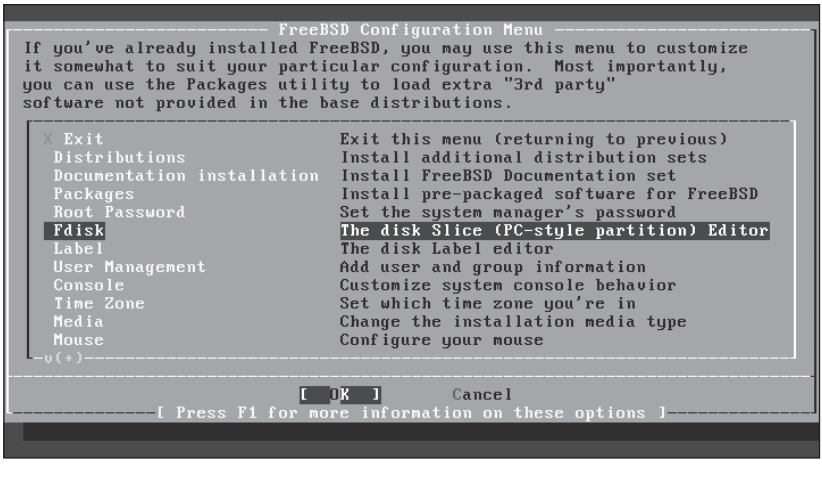

Рис. 7.3. Выберите команду Fdisk

Затем конфигуратор попросит вас выбрать диск, разметку которого вы хотите произвести. Далее вы увидите интерфейс, подобный тому, который мы использовали при установке операционной системы в главе 2. Чтобы здесь не повторяться, отправляю вас назад в прошлое — в главу 2. Вы можете использовать автоматическую разметку — будет создан один BSD-слайс размером на весь жесткий диск — как раз то, что нам и нужно. Когда все будет готово, нажмите W для записи изменений, а затем — Q.

А вот после выхода из Fdisk конфигуратор sysinstall предложит вам записать загрузчик на ваш новый жесткий диск. Не нужно этого делать. Выберите None для отказа от установки загрузчика. Вы вернетесь в меню, изображенное на рис. 7.3. Далее вам нужно выбрать команду Label для создания BSD-разделов. Интерфейс этой программы тоже описан в главе 2. Вот только вам не нужно использовать автоматическую разметку, поскольку ни раздел a (корневая файловая система), ни раздел b (подкачка) нам не нужны. Следовательно, нажмите C для создания BSD-раздела, по умолчанию будет создан раздел размером с весь жесткий диск — здесь решать вам: создать один большой раздел или создать несколько меньшего размера. После ввода размера раздела (можно указать размер или в блоках или в мегабайтах, например 10240M) нужно выбрать тип раздела — FS или SWAP. Нам не нужен раздел подкачки, поэтому нужно выбрать FS. В конечном итоге будет создан раздел d (так как имена a, b, c зарезервированы) или несколько разделов, если вам не нужен один большой раздел.

При создании раздела вы можете также указать точку монтирования. Однако каталог, который вы укажете, должен быть создан на момент вызова sysinstall. Если вы не знаете пока, куда будете монтировать разделы, просто нажмите Enter. Нажмите W для записи изменений и Q для выхода.

Затем вы можете просмотреть на результат вашего творчества:

```
# bsdlabel <имя слайса>
```

Например:

```
# bsdlabel /dev/ad6s1
```

Если вы создали всего один раздел, вывод будет примерно таким:

```
# /dev/ad6s1:
8 partitions:
#        size   offset    fstype   [fsize bsize bps/cpg]
  c: 16776522        0    unused        0     0         # "raw" part, don't edit
  d: 16776522        0    4.2BSD        0     0     0
```

Вы увидите список всех созданных вами BSD-разделов. Осталось их только прописать в /etc/fstab и выполнить команду mount -a.

Хотя конфигуратор sysinstall создает файловые системы, вы должны знать, какие команды используются в FreeBSD для создания файловых систем:

- newfs — создает файловую систему UFS/UFS2;
- mkfs.ext2, mkfs.ext3, mkfs.ext4 — создают файловые системы ext2, ext3 и ext4 соответственно.

Загрузка системы

8

8.1. Процесс загрузки Linux

8.1.1. Загрузчик и ядро Linux

При включении компьютера запускается процедура самотестирования — POST (Power On Self Test). После этого управление передается загрузчику. Его задача — отыскать загрузчик операционной системы и передать ему управление. Обычно поиск начинается с главной загрузочной записи компакт-диска (Master Boot Record, MBR), хотя порядок загрузки задается опцией в SETUP (опция обычно называется Boot Sequence).

Основные загрузчики Linux — LILO, GRUB и GRUB2. LILO уже канул в лету и практически не используется, его дальнейшая разработка приостановлена. Во многих современных дистрибутивах используется загрузчик GRUB, несмотря на наличие второй версии — GRUB2. Но статус экспериментального для GRUB2 пока еще никто не отменял (хотя он уже вовсю используется в Ubuntu). В этой книге мы будем использовать обычный GRUB, который используется в современных версиях Mandriva, Fedora, openSUSE и других дистрибутивах.

Итак, из MBR будет запущен загрузчик GRUB. GRUB в соответствии с файлом `/boot/grub/menu.lst` отобразит меню операционных систем. Если пользователь выберет Windows, тогда управление будет передано загрузчику NTLDR, который и загрузит Windows. А вот если пользователь выберет Linux, то загрузчик запустит ядро Linux.

Ядро Linux — это обычная программа, следовательно, ей можно передать параметры. Вот наиболее часто использующиеся параметры ядра:

- `root=устройство` — задает раздел Linux, содержащий корневую файловую систему. Если не передать этот параметр, Linux не загрузится.
- `init=программа` — задает программу, выполняющую инициализацию Linux. Обычно это `/sbin/init`, но в некоторых случаях можно указать другую программу, если нельзя выполнить инициализацию системы обычным образом.

- single — загрузка системы в однопользовательском режиме (используется в аварийных ситуациях).
- noapic — если есть проблемы с APIC.
- splash=тип_заставки — раньше при загрузке Linux пользователи видели диагностические сообщения ядра. В современных дистрибутивах разработчики Linux отказались от вывода этих сообщений в пользу красивой графической заставки, чтобы не шокировать начинающих пользователей. Управлять заставкой можно с помощью параметра ядра splash, который может принимать три значения:
  - splash=0 — заставка отключена, выводятся сообщения ядра (некрасиво, зато информативно);
  - splash=verbose — сообщения ядра выводятся на красивом графическом фоне (и красиво и информативно). Не знаю как вам, а мне больше всего нравится именно этот режим;
  - splash=silent — сообщения ядра не выводятся, только графическая заставка (красиво, но не информативно). Используется по умолчанию.

Сообщения ядра всегда (вне зависимости от параметра splash) протоколируются в файле /var/log/dmesg, просмотреть который можно с помощью команды:

```
# dmesg | less
```

При загрузке ядро, помимо всего прочего, выполняет две очень важные операции: монтирование корневой файловой системы и запуск программы инициализации системы, которая выполняет всю последующую загрузку системы.

8.1.2. Система инициализации init

Для Linux разработано несколько систем инициализации, но классическая init до сих пор продолжает использоваться во многих современных дистрибутивах, именно поэтому она и будет рассмотрена в этой книге.

С одной стороны, это довольно «древняя» система. Существуют более новые системы инициализации, работающее более эффективно, чем init (читайте — сокращают время загрузки системы). А с другой стороны, init — проверенное временем надежное решение, известное всем администраторам Linux, поэтому об init еще не скоро забудут.

Первым делом init определяет уровень запуска системы, который хранится в файле /etc/inittab. Откройте этот файл, и вы найдете в нем строку:

```
id:5:initdefault:
```

Не нужно быть гением, чтобы понять, что сейчас система по умолчанию запускается на пятом уровне. Всего есть шесть уровней запуска:

- 0 — останов системы (данный уровень запуска нельзя использовать в качестве уровня запуска по умолчанию);
- 1 — однопользовательский режим;
- 2 — многопользовательский режим, но без сети;
- 3 — многопользовательский режим с поддержкой сети;

- 4 — не используется;
- 5 — то же, что и 3, но производится загрузка графического интерфейса;
- 6 — перезапуск системы (данный уровень запуска нельзя использовать в качестве уровня запуска по умолчанию).

Понятно, что если вместо 5 подставить 3, то система будет запущена без графического интерфейса. Если система уже запущена на пятом уровне, а вам нужно перейти на третий уровень, тогда нужно использовать команду:

```
# /sbin/init 3
```

Кроме уровня запуска по умолчанию в файле /etc/inittab задаются и другие параметры инициализации систем (листинг 8.1).

Листинг 8.1. Файл /etc/inittab с комментариями на русском языке

```
#
# Автор:     Денис Колисниченко, <dhsilabs@dkws.org.ua>

# Уровень запуска по умолчанию. Можно использовать следующие уровни:
#   0 - останов системы
#   1 - однопользовательский режим
#   2 - многопользовательский режим без сети
#   3 - многопользовательский режим с поддержкой сети
#   4 - не используется
#   5 - X11
#   6 - перезагрузка
#
id:5:initdefault:

# Сценарий, выполняющий основные действия по инициализации системы
si::sysinit:/etc/rc.d/rc.sysinit

l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6

# Команда, которая будет выполнена при нажатии CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now

# Реакция на сигнал PowerFail (отключение питания) от UPS
# Система подождет 2 минуты, а потом начнет разгрузку (завершение работы)
pf::powerfail:/sbin/shutdown -f -h +2 "Power Failure; System Shutting Down"

# Если в течение двух минут питание будет восстановлено, нужно отменить
# предыдущую команду
pr:12345:powerokwait:/sbin/shutdown -c "Power Restored; Shutdown Cancelled"

# Запуск gettys на стандартных уровнях запуска на консолях 1-6
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
```

```
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6

# Оболочка для однопользовательского режима
~~:S:wait:/bin/sh
```

Выбрать уровень запуска можно или путем редактирования файла /etc/inittab.

Сценарии системы инициализации хранятся в каталоге /etc/rc.d. В этом каталоге вы найдете ряд подкаталогов и три файла:

- rc — отвечает за запуск/останов сервисов при изменении уровня запуска;
- rc.local — запускается после всех сценариев инициализации, в этот файл принято добавлять команды, которые должны запускаться автоматически;
- rc.sysinit — основной файл инициализации, запускается первым вне зависимости от уровня запуска.

В каталоге /etc/rc.d/init.d находятся сценарии запуска сервисов (служб). Например, сценарий /etc/rc.d/init.d/crond управляет запуском и остановом сервиса `crond` — планировщика заданий.

Каждому сценарию из каталога init.d можно передать один из трех параметров (хотя параметров может быть больше, это зависит от самого сервиса, например часто используется параметр `reload` для обновления конфигурации сервиса без его перезапуска):

- `start` — запуск сервиса;
- `restart` — перезапуск сервиса;
- `stop` — останов.

А теперь начинается самое интересное. В каталогах /etc/rc.d/rcN.d (где N — цифра от 0 до 6, соответствующая уровню запуска) находятся ссылки на сценарии сервисов, которые нужно выполнить при переходе системы на тот или иной уровень запуска.

Зайдите в один из каталогов, например в rc5.d. В нем вы найдете ссылки двух видов:

- KчислоНазвание_сервиса;
- SчислоНазвание_сервиса.

Возьмем две ссылки: `K49netconsole` и `S03iptables`. Если имя ссылки начинается с буквы K (первая буква слова «kill»), при переходе на данный уровень запуска (в нашем случае — 5) система инициализации должна выполнить заданный сервис (в нашем случае — netconsole) с параметром stop:

```
/etc/rc.d/init.d/netconsole stop
```

Если же имя ссылки начинается буквой S (первая буква в слове «start»), система выполнит соответствующий сервис (iptables) с параметром start:

```
/etc/rc.d/init.d/iptables start
```

Число в названии ссылки задает приоритет, например сервис S03iptables будет выполнен раньше, чем S04acpi.

С помощью ссылок в каталогах rcN.d система инициализации управляет запуском и остановом сервисов при переходе на разные уровни запуска. Понятно, что на уровнях 3 и 5 S-ссылок будет больше, чем K-ссылок, зато на уровнях останова (0 и 6) больше будет K-ссылок.

Вы вряд ли будете создавать или удалять ссылки в каталогах rcN.d вручную, поскольку для этого обычно используют специальные графические конфигураторы, например drakxservices в Linux Mandriva (рис. 8.1) или system-config-services (рис. 8.2) в Fedora. В openSUSE для управления сервисами используется главный конфигуратор Yast. В Mandriva есть еще один полезный конфигуратор, позволяющий выбрать уровень запуска по умолчанию, — drakboot (рис. 8.3). Этот же конфигуратор позволяет настроить автоматический вход пользователя.

Кнопка Настройка в конфигураторе system-config-services позволяет выбрать уровни запуска, на которых будет запускаться тот или иной сервис (рис. 8.4).

FreeBSD мне нравится тем, что сетевые сервисы в ней запускаются именно те, которые нужны вам, а не все подряд. Вам понадобился, скажем, Apache, вы добавили соответствующую строку запуска в файл /etc/rc.conf. А вот в Linux сразу после установки запущено множество сервисов, которые нужны всем и сразу, но это не означает, что они нужны вам. Отключив неиспользуемые сервисы, вы снижаете нагрузку на процессор, уменьшаете потребление оперативной памяти и сокращаете время загрузки системы (чем больше сервисов, тем медленнее загружается система). Названия и активность (включен или выключен по умолчанию)

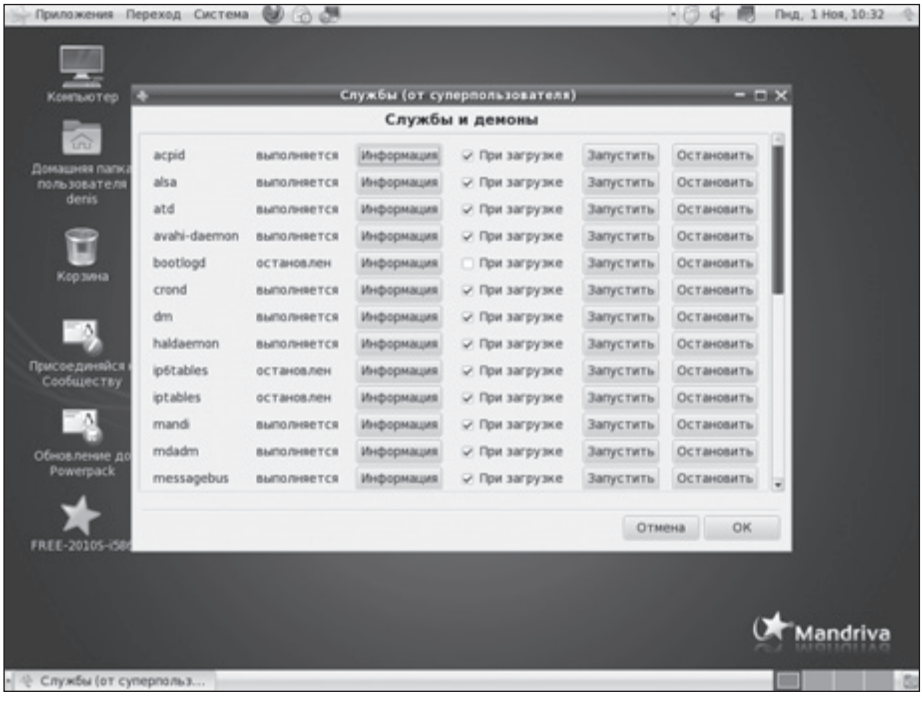

Рис. 8.1. Конфигуратор drakxservices

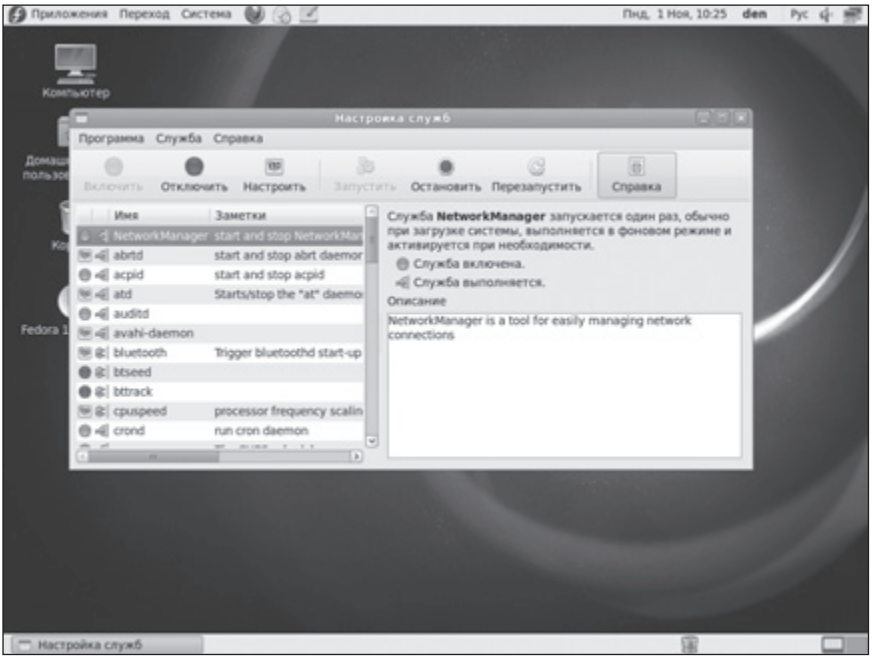

Рис. 8.2. Конфигуратор system-config-services

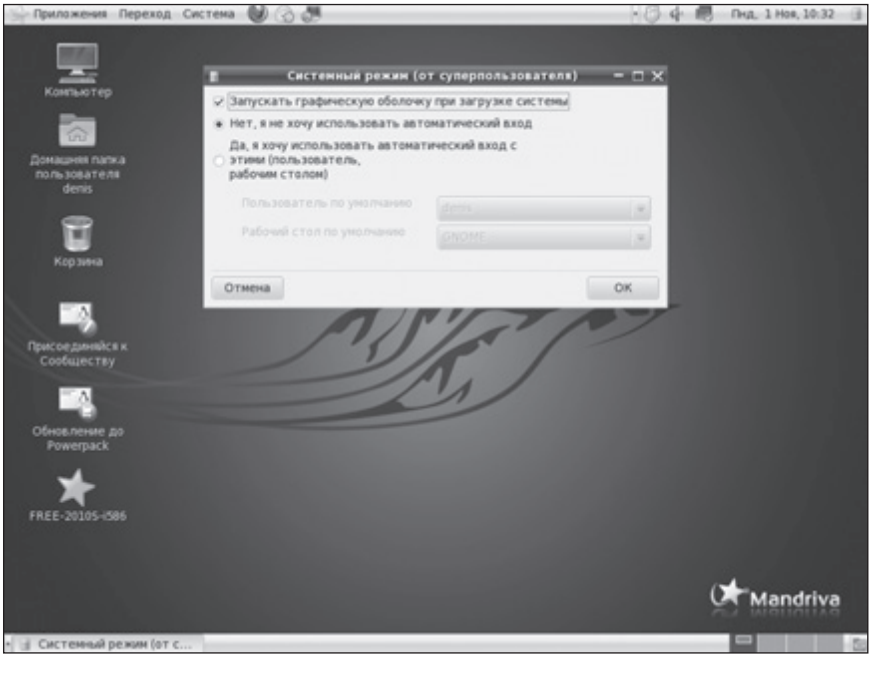

Рис. 8.3. Конфигуратор drakboot

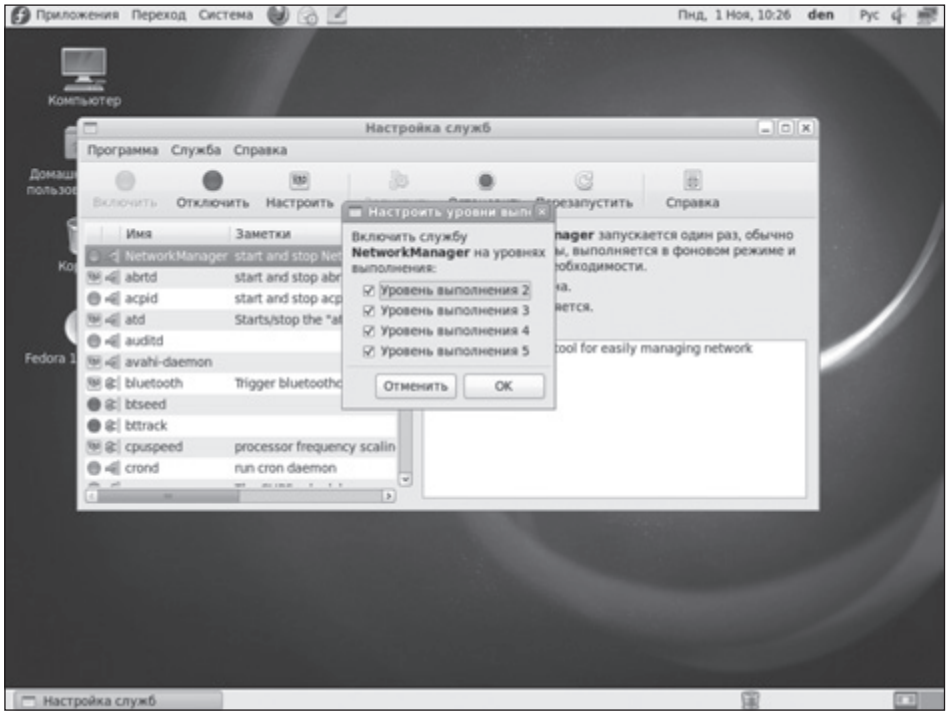

Рис. 8.4. Настройка сервиса

сервисов зависит от дистрибутива, но, ориентируясь на два базовых дистрибутива Fedora и Mandriva, можно отключить следующие сервисы:

- abrt (Automatic Bug Reporting Tool) — занимается сбором информации о сбое системы для отправки этой информации разработчикам дистрибутива, отключайте смело, даже не задумываясь;
- apmd — управляет питанием, нужен только на ноутбуках, на стационарных компьютерах можно отключить;
- anacron, atd, crond — планировщики заданий (позволяют запустить указанную вами программу в выбранное время), обычно нужны только на сервере, а на рабочих станциях можно смело выключить;
- avahi-daemon — реализует стек mDNS (Multicast DNS), в большинстве случаев можно отключить;
- auditd — система аудита Linux. Можно смело отключить, в этом случае все события будут отправлены на демон протоколирования syslog, а вот его отключать не нужно. Вообще так даже удобнее — пусть сбором событий занимается один сервис, а не два;
- bluetooth — обеспечивает поддержку Bluetooth. Если у вас нет Bluetooth-адаптера или вы не собираетесь использовать Bluetooth, отключите этот сервис;
- btseed — нужен для торрент-клиента BitTorrent, если вы его не используете, выключите этот сервис;

- cups* — система печати CUPS (Common Unix Printing System). На сервере, где не планируется печать ни на локальном, ни на сетевом принтере, можно выключить;
- dnsmasq — запускает кэширующий DNS-сервер. Самое интересное, что данный сервер нужен далеко не всем пользователям (поскольку в сети обычно уже есть такой сервер), а в некоторых дистрибутивах сервис dnsmasq запускается автоматически и расходует системные ресурсы. Отключите его, если, конечно, он вам не нужен;
- hidd — демон HIDD (Human Interface Device Daemon) добавляет поддержку клавиатур, манипуляторов и других устройств ввода информации, работающих по Bluetooth. У меня таких нет и не было, поэтому я этот сервис отключил;
- hddtemp — позволяет получить информацию о температуре жесткого диска. Постоянно этот сервис не нужен, если появятся подозрения, что жесткий диск греется, тогда можно будет запустить его вручную;
- httpd (или apache) — веб-сервер Apache, нужен только на соответствующем сервере, можно отключить на рабочих станциях;
- isdn — обеспечивает поддержку ISDN-линии, но совершенно бесполезен, если у вас нет таковой;
- ip6tables — брандмауэр iptables для протокола IPv6, но поскольку этот протокол еще не используется, то сервис можно выключить;
- lm_sensors — позволяет получить разную информацию об аппаратных средствах, например температуру процессора, материнской платы, скорость вращения вентилятора и т. д. Однако не все «железо» поддерживает этот сервис, поэтому lm_sensors обычно можно выключить без потери функциональности для системы;
- mandi — занимается мониторингом сети, нужен только на сервере и то не всегда;
- mdadm — поддержка RAID-массивов, если у вас таких нет, отключайте;
- mdmonitor — тоже относится к RAID, можно выключить, если RAID не используется;
- partmon — проверяет, есть ли на жестком диске свободное место. А не выключить ли нам его?
- pcmcia (pccsd) — обеспечивает поддержку PCMCIA-карт. На стационарном компьютере можно выключить;
- sendmail (или postfix) — запускает агент отправки почты (MTA), нужен на сервере, но не нужен на рабочей станции;
- smartd — обеспечивает поддержку S.M.A.R.T-устройств. Нет таких устройств? Значит, выключите сервис;
- snmp* — поддержка протокола управления сетью SNMP (Simple Network Management Protocol);
- sshd — обеспечивает удаленный доступ к консоли машины, на рабочих станциях можно отключить, на сервере (при настройке sshd и необходимости) можно оставить включенным;

После отключения ненужных сервисов нужно перезагрузить компьютер, чтобы оценить, как сократилось время загрузки:

```
# reboot
```

Что же касается Debian и Ubuntu, то лишних сервисов там практически нет. В Debian и Ubuntu (до версии 8.10) для управления сервисами используется конфигуратор services-admin. Начиная с версии 8.10 нужно использовать конфигуратор bum, который, однако, не установлен по умолчанию. Для его установки нужно ввести команду:

```
sudo apt-get install bum
```

8.1.3. Настройка загрузчика GRUB

Конфигурационный файл GRUB

Основной конфигурационный файл называется menu.lst и находится он в каталоге /boot/grub. В некоторых дистрибутивах данный файл называется grub.conf, но чаще все-таки menu.lst.

Конфигурационный файл содержит инструкции по загрузке той или иной операционной системы, а также параметры самого загрузчика. Нужно быть очень внимательным при редактировании этого файла. Поэтому не спешите, и прежде чем перезагрузить компьютер, убедитесь, что все правильно. Иначе при следующем запуске меню загрузчика может и не быть.

Начнем с описаний операционных систем. Вот типичное описание операционной системы.

```
titleMy Linux
root(hd0,5)kernel/boot/vmlinuz-2.6.30-15-generic root=UUID=1f049af9-2bdd-43bf-a16c-
ff5859a4116a ro quiet splash locale=ru_RUinitrd/boot/initrd.img-2.6.30-15-genericquiet
```

Описание каждой ОС начинается с директивы title. После служебного слова title может следовать произвольная строка — ее вы увидите в меню загрузчика. Директива root задает имя корневой файловой системы, точнее имя раздела, на котором она находится. Тут нужно запомнить два момента: директиву root желательно использовать только для описания корневого раздела Linux и UNIX-подобных ОС. Для задания загрузочного раздела Windows используется директива rootnoverify. Второй момент заключается в том, что GRUB немного не так именует разделы жесткого диска. Мы привыкли, что «первичный мастер» называется /dev/hda. Соответственно, первый раздел на нем будет называться /dev/hda1. Как правило, это диск C, то есть загрузочный раздел Windows. В GRUB все иначе. Наименование жесткого диска и раздела записывается так:

```
(hdN,M)
```

N — это номер жесткого диска, нумерация начинается с 0. А M — номер раздела на жестком диске, нумерация тоже начинается с 0. Исходя из этого, раздел /dev/hda1 будет именоваться как hd(0,0). Первый 0 задает первичный мастер (/dev/hda), а второй — первый (пусть это слово вас не смущает — нумерация с 0) раздел на этом диске. Вот как будет выглядеть описание загрузки Windows:

```
titleMicrosoft Windowsrootnoverify(hd0,0)makeactivechainloader+1
```

Директива rootnoverify задает загрузочный раздел Windows — /dev/hda1. Директива makeactive делает раздел /dev/hda1 активным. Для Windows XP (как и для 2000, Vista) — это непринципиально, но обязательно для Windows 9x, особенно если Windows установлена на неактивном разделе.

ПРИМЕЧАНИЕ

Даже если ваш жесткий диск называется /dev/sda, а не /dev/hda, то его имя в GRUB от этого не изменится — это будет по-прежнему hd0. hd* — это внутреннее имя GRUB, ничего не имеющего общего с системным именем устройства /dev/?d?. В новых дистрибутивах даже IDE-диски называются /dev/sd?. У такого решения есть и преимущества, и недостатки. Однако мне все равно больше нравилась старая схема, когда IDE-диски назывались hd?, а SCSI/SATA — sd?.

ПРИМЕЧАНИЕ

В таблице разделов есть специальный атрибут, указывающий на активность раздела. Активным может быть только один раздел. Ранние версии Windows умели загружаться только с активных разделов.

Директива chainloader нужна для операционных систем, которые поддерживают цепочечную загрузку. Windows как раз и является такой ОС.

Вернемся к описанию загрузочного образа Linux. Директива kernel описывает параметры ядра. Один из главных параметров — это параметр root. Он задает имя раздела, содержащего корневую файловую систему. В качестве имени можно указывать идентификатор раздела (что и делает по умолчанию Ubuntu) или его имя. В моей системе идентификатор 1f049af9-2bdd-43bf-a16c-ff5859a4116a соответствует разделу /dev/sda6, поэтому я могу записать строку kernel так:

```
kernel/boot/vmlinuz-2.6.30-15-generic root=/dev/sda6 ro quiet splash locale=ru_RU
```

Как узнать, какой идентификатор соответствует какому разделу? А зачем это вам нужно? Все равно данные идентификаторы вам не придется изменять, а если и нужно, то гораздо проще указать имя раздела, чем его идентификатор.

Выбор операционной системы по умолчанию

Вы хотите установить Windows в качестве операционной системы по умолчанию? Тогда вам нужно изменить значение директивы default конфигурационного файла. Давайте рассмотрим пример реального конфигурационного файла. Это мой файл menu.lst. Конечно, я удалил из него все комментарии, ненужные в данный момент опции и все описания операционных систем после Windows. Зачем? Для экономии места в книге и вашего времени. Когда я вставил свой menu.lst в окно OpenOffice Writer, то размер этой главы увеличился на 10 страниц. Теперь, надеюсь, вы понимаете, почему мы будем работать с сокращенной версией этого файла (листинг 8.2).

Директива default задает номер загрузочной метки по умолчанию. Нумерация начинается с 0. Я выделил жирным все загрузочные метки, чтобы вам было проще считать. Итак, считаем: 0, 1, 2, 3, 4. Загрузочная метка Windows пятая в списке, следовательно, ее номер 4. Вот его и нужно записать в качестве значения директивы default.

Листинг 8.2. Файл /boot/grub/menu.lst

```
default    4
timeout    10

title      My Linux
root       (hd0,5)
kernel     /boot/vmlinuz-2.6.30-15-generic root=UUID=1f049af9-2bdd-43bf-a16c-ff5859a4116a ro
           quiet splash
locale=ru_RU
initrd     /boot/initrd.img-2.6.30-15-generic
quiet
savedefault

title      My Linux (recovery mode)
root       (hd0,5)
kernel     /boot/vmlinuz-2.6.30-15-generic root=UUID=1f049af9-2bdd-43bf-a16c-ff5859a4116a ro
           single
initrd     /boot/initrd.img-2.6.30-15-generic

title      Memory test
root       (hd0,5)
kernel     /boot/memtest86+.bin
quiet

title       Other operating systems:
root

title      Microsoft Windows
root       (hd0,0)savedefaultmakeactivechainloader    +1
```

Директива timeout задает количество секунд, по прошествии которого будет загружена операционная система по умолчанию.

Цвета и фон загрузчика

Меню начального загрузчика, мягко говоря, выглядит невесело. Попробуем исправить это. Для начала раскомментируйте параметр color. Он задает цветовую схему загрузчика. Если знаете, как называются цвета по-английски, можете поэкспериментировать с ними. А если нет, тогда оставьте как есть — станет намного симпатичнее:

```
# Pretty colourscolor cyan/blue white/blue
```

Вы также можете задать картинку в качестве фона загрузчика. Для этого берем картинку любого формата, например JPEG. Желательно, чтобы размеры картинки были 640 × 480. Изображение нужно преобразовать в формат XPM. Это можно сделать с помощью GIMP или с помощью программы convert. Кстати, если будете преобразовывать изображение в GIMP, не забудьте изменить его размер — 640 × 480. Если вы хотите использовать приложение convert, тогда введите команду:

```
convert source.jpg -colors 14 -resize 640x480 dest.xpm
```

ПРИМЕЧАНИЕ

Программа convert есть в пакете imagemagick, поэтому, прежде чем ее использовать, нужно установить данный пакет

Откройте файл menu.lst и добавьте в его начало строку:

```
splashimage=(hd0,5)/boot/grub/dest.xpm
```

Результирующий файл dest.xpm нужно скопировать в каталог /boot/grub на разделе /dev/hda6 (имя раздела измените на свое). Сохраните файл menul.st и перезагрузитесь.

8.1.4. Планировщики заданий

Планировщик cron

Довольно часто приходится периодически запускать одни и те же задачи. Конечно, если вам нужно всего лишь раз в неделю запустить обновление антивируса, можно и не ломать себе голову настройкой планировщика. А вот если у вас давно сформировался список задач, которые вы выполняете чуть ли не каждый день и вы просто не хотите в один момент забыть выполнить какую-то программу, тогда вам поможет планировщик заданий.

Запустите конфигуратор drakxservices. Если просмотреть список служб, то вы найдете в нем два планировщика заданий — crond и atd. Первый используется для создания долговременного списка заданий. Второй следует использовать, если вам нужно не забыть в определенное время (на протяжении дня) выполнить какую-то команду. Например, команду выключения компьютера в 23:55.

Откройте файл /etc/crontab — это основной файл планировщика crond. Типичный файл crontab представлен в листинге 8.3

Листинг 8.3. Файл /etc/crontab

```
SHELL=/bin/shPATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/
# run-parts01 * * * * root nice -n 19 nice run-parts --report /etc/cron.hourly
02 4 * * * root nice -n 19 nice run-parts --report /etc/cron.daily
22 4 * * 0 root nice -n 19 nice run-parts --report /etc/cron.weekly
42 4 1 * * root nice -n 19 nice run-parts --report /etc/cron.monthly
```

В каталоге /etc/cron.daily находятся сценарии, которые должны запускаться ежедневно (рис. 8.5). В каталоге cron.weekly — те, которые должны выполняться еженедельно, а в каталоге cron.monthly — ежемесячно.

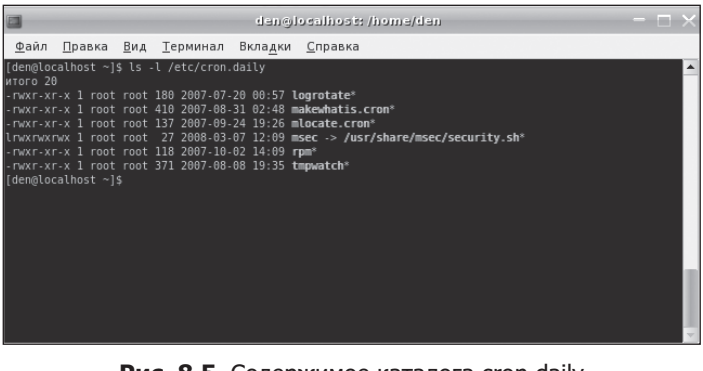

Рис. 8.5. Содержимое каталога cron.daily

Чтобы ваш сценарий выполнялся ежедневно/еженедельно/ежемесячно, его нужно поместить в соответствующий каталог.

Вообще формат файла crontab следующий:

```
минута час день месяц день_недели пользователь команда
```

Назначение первых пяти полей ясно из названия, шестое поле — имя пользователя, от имени которого будет выполнена команда. Поля разделяются пробелами или символами табуляции. Рассмотрим несколько примеров:

```
# Запускать программу /bin/command1 в 4:05 каждое воскресенье
5 4 * * sun root /bin/command1
# Запускать программу /bin/command2 в 9:00 каждый рабочий день (1-5)
0 9 * * 1-5 root /bin/command2
# Запускать в 9:00 первого числа каждого месяца command3
0 9 1 * * root /bin/command3
```

Напомню, что команды из расписания crontab будут выполнены только в том случае, если соответствующая служба (crond) запущена.

Планировщик atd

Планировщик задач atd использовать намного проще. Просто введите следующую команду:

```
at время [дата]
```

Если дата не указана, то будет считаться, что указанные вами команды нужно выполнить в указанное время сегодня.

После того как вы нажмете Enter, вам нужно будет ввести список команд, которые нужно выполнить в указанное время. Для завершения ввода нужно нажать Ctrl + D.

Время указывается в формате AM/PM, то есть для выполнения команды в 23:00 нужно ввести команду:

```
at 11pm
```

А затем ввести нужную команду. Минуты указываются через :, например

```
at 11:55pm
```

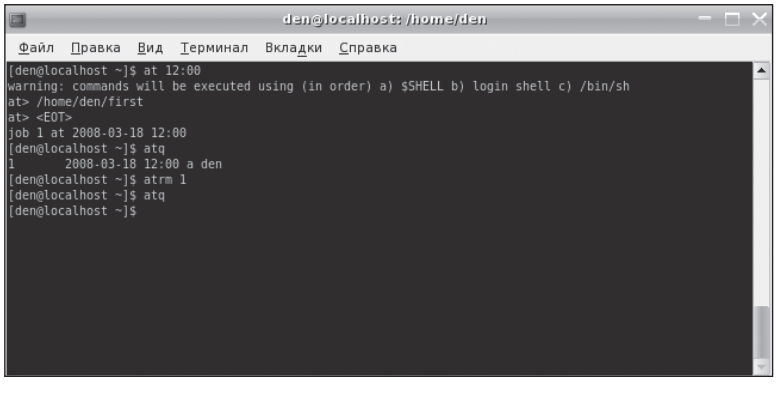

Рис. 8.6. Использование планировщика atd

Для просмотра очереди заданий введите команду atq. Если вам нужно удалить задание, тогда используйте команду:

```
atrm <номер задания>
```

Номер задания можно узнать с помощью команды atq. Рассмотрим рис. 8.6. Сначала мы добавляем в очередь заданий команду /home/den/first, которая должна быть выполнена в 12:00. Затем мы просматриваем список заданий, номер нашего задания — 2. После этого мы удаляем задание с этим номером.

8.2. Процесс загрузки BSD

8.2.1. Загрузка FreeBSD

В отличие от современных дистрибутивов Linux, где, чтобы увидеть сообщения ядра при загрузке вместо графической заставки, нужно передать ядру специальный параметр, во FreeBSD вы видите все сообщения, выводимые ядром при загрузке. Понятно, что прочитать их вы не успеете, но это не беда — есть команда dmesg. По большому счету так и должно быть — процесс загрузки FreeBSD похож на процесс загрузки той самой BSD, разработанной в 70–80-х годах прошлого века. А вот процесс загрузки современных дистрибутивов Linux похож на... Windows. Видимо, чтобы не отпугивать начинающих пользователей. Но это все равно, что «Хаммер» покрасить в розовый цвет и «обуть» в хромированные диски (правда, видел и такие экземпляры). Именно поэтому в моем дистрибутиве Denix нет ни индикатора загрузки, ни текстовой заставки — вы видите все сообщения, выводимые ядром.

С технической точки зрения, процесс загрузки FreeBSD похож на процесс загрузки Linux. Загрузчик операционной системы находит в MBR загрузчик FreeBSD и передает ему управление, после чего вы видите меню выбора операционной системы:

```
F1FreeBSD
F6 PXE
Boot: F1
```

Нажав F1, вы загрузите FreeBSD. Вариант, запускаемый нажатием F6, запускает PXE (Preboot Execution Environment, произносится «пикси»). PXE — это среда загрузки компьютера по сети, используется для построения бездисковых станций. PXE не рассматривается в этой книге.

После нажатия F1 увидите варианты загрузки FreeBSD (рис. 8.7)

Как и в случае с Linux, есть две стадии загрузчика. Первая стадия (boot0) — это запуск первого загрузочного блока в MBR. Именно на первой стадии отображается меню выбора операционной системы. Вторая стадия — запуск второго загрузочного блока (boot1). Его задача — найти и передать управление третьему загрузочному блоку (boot2). Третий загрузочный блок отображает варианты загрузки FreeBSD. Основной файл конфигурации загрузчика, по которому и строится меню, отображенное на рис. 8.7, называется /boot/loader.conf.

Рис. 8.7. Варианты загрузки FreeBSD

Рассмотрим варианты загрузки FreeBSD:

- Boot FreeBSD — обычная загрузка FreeBSD, в большинстве случаев нужно выбрать этот вариант;
- Boot FreeBSD with ACPI disabled — загрузка FreeBSD с отключенным ACPI, может понадобиться на некоторых аппаратных конфигурациях, например если у вас процессор AMD64 и видеокарта от NVIDIA;
- Boot FreeBSD in Safe Mode — загрузка системы в безопасном режиме (если после настройки что-то пошло не так);
- Boot FreeBSD in single user mode — загрузка в однопользовательском режиме, подходит для глубокой настройки системы, когда не нужно, чтобы машина работала в сети и принимала регистрацию других пользователей. Подходит также для восстановления системы после сбоя.
- Boot FreeBSD with verbose logging — расширение протоколирования при загрузке, поможет разобраться с проблемой, если вы не знаете ее причину;
- Escape to loader prompt — перейти в командную строку загрузчика;
- Reboot — перезагрузить компьютер.

Задача загрузчика — передать управление ядру операционной системы. Все остальное — уже проблемы ядра. После загрузки ядро передает управление системе инициализации, которая проверяет и монтирует файловую систему, а также файловые системы, указанные в /etc/fstab, и запускает сервисы, «прописанные» в /etc/rc.conf, настраивает сеть. Конечно, система инициализации выполняет не только эти действия, но о них мы подробно поговорим позже. Если в процессе проверки файловой системы будет обнаружена ошибка, вам будет предложено перейти в однопользовательский режим для проверки файловой системы (используется команда fsck). В случае отсутствия проблем в конечном итоге вы увидите приглашение войти в систему.

8.2.2. Система инициализации FreeBSD

В каталоге /etc/rc.d находятся сценарии загрузки системы. Каждый такой сценарий запускает тот или иной сервис, например /etc/rc.d/sshd запускает SSH-сервер, а сценарий /etc/rc.d/inetd — суперсервер inetd.

Сценарии из каталога /etc/rc.d могут не только запускать сервисы, но и останавливать, перезагружать. Одним словом, они позволяют управлять сервисами. Иногда для управления сервисом предоставляется отдельная программа, например для управления сервером Apache можно использовать программу apachectl. Что именно использовать — программу управления или сценарий из /etc/rc.d, — значения не имеет — результат будет таким же. В большинстве случаев сценарии из /etc/rc.d не требуют редактирования. Просто вы теперь знаете, где они находятся и для чего используются, а вот редактировать их незачем.

В каталоге /etc/rc.d много сценариев, откуда система знает, какие нужно загружать, а какие нет? Информацию об этом она получает из файла /etc/rc.conf. Например, если в этом файле есть строка apache22_enable="YES", то система выполнит команду /etc/rc.d/apache22 start или /usr/local/etc/rc.d/apache22 start — в зависимости от расположения сценария apache22. Система ищет сценарии запуска в каталогах /etc/rc.d и /usr/local/etc/rc.d. В последний, как правило, помещаются пользовательские сценарии запуска, а в /etc/rc.d — общесистемные.

Если вам интересно, в каком порядке будут запущены сценарии запуска системы, тогда выполните команду:

```
rcorder /etc/rc.d/*
```

Данная команда выводит порядок запуска сценариев, но не может изменить его. Самый главный сценарий инициализации системы называется /etc/rc. Этот сценарий читает файл /etc/rc.conf и выполняет инициализацию системы — запускает необходимые сервисы.

Пример файла /etc/rc.conf вы найдете в каталоге /etc/defaults. В нем описаны самые разнообразные параметры, которые только могут быть в файле /etc/rc.conf (см. листинг 8.3). Для большего удобства все комментарии в листинге 8.4 переведены мною на русский язык.

Листинг 8.4. Файл /etc/defaults/rc.conf

```
#!/bin/sh

##############################################################
###  Параметры загрузки  #####################################
##############################################################

rc_debug="NO"               # YES = включить режим отладки
rc_info="NO"                # YES = отображать информационные
                            # сообщения при загрузке

rcshutdown_timeout="30"     # Сколько секунд ждать перед завершением rc.shutdown
early_late_divider="FILESYSTEMS"    # Скрипт, который разделяет раннюю и
                            # позднюю стадии загрузки. Не изменяйте это значение

swapfile="NO"               # Имя файла подкачки или значение NO
```

продолжение ⇩

Листинг 8.4 *(продолжение)*

```
apm_enable="NO"             # YES = включение функции APM BIOS
apmd_enable="NO"            # YES = запуск демона apmd для обработки APM-событий
apmd_flags=""               # Параметры демона apmd
ddb_enable="NO"             # YES = запуск ddb-сценариев при загрузке

ddb_config="/etc/ddb.conf"  # Конфиг для ddb
devd_enable="YES"           # YES = запуск демона devd
devd_flags=""               # Параметры для devd.
kldxref_enable="NO"         # Создавать файлы linker.hints с kldxref(8).
kldxref_clobber="NO"        # Перезаписывать старые файлы linker.hints при загрузке
kldxref_module_path=""      # Путь kern.module_path.
powerd_enable="NO"          # YES = запуск демона управления питанием
powerd_flags=""             # Параметры powerd
tmpmfs="AUTO"               # YES = всегда создавать mfs /tmp
tmpsize="20m"               # Размер mfs /tmp (mfs - memory filesystem)
tmpmfs_flags="-S"           # Параметры mfs
varmfs="AUTO"               # YES = чтобы создавать mfs /var
varsize="32m"               # Размер mfs /var (размер файловой системы в памяти)
varmfs_flags="-S"           # Параметры для монтирования mfs /var
cleanvar_enable="YES"       # Очищать каталог /var (для экономии места)
local_startup="/usr/local/etc/rc.d" # Пользовательские сценарии конфигурации
script_name_sep=" "         # Измените разделитель, если ваши сценарии конфигурации
                            # содержат пробелы
rc_conf_files="/etc/rc.conf /etc/rc.conf.local"

# Поддержка ZFS
zfs_enable="NO"             # YES = ZFS-разделы будут монтироваться автоматически

# Экспериментальные опции! Не рекомендуются на стабильной системе!
# Устройства с шифрованием (GEOM Based Disk Encryption)
gbde_autoattach_all="NO"    # YES = авт. монтиров. gbde-устройств из fstab
gbde_devices="NO"           # Устройства gdbe или значение NO
gbde_attach_attempts="3"    # К-во попыток подключения gdbe-устройства
gbde_lockdir="/etc"         # Каталог для файлов блокировок gdbe

# Устройства с шифрованием GELI. Параметры такие же, как и для GEOM
geli_devices=""
geli_tries=""
geli_default_flags=""
geli_autodetach="YES"

# Пример определения GELI-устройства
#geli_devices="da1 mirror/home"
#geli_da1_flags="-p -k /etc/geli/da1.keys"
#geli_da1_autodetach="NO"
#geli_mirror_home_flags="-k /etc/geli/home.keys"

# Опции для зашифрованного с помощью GELI своп-раздела
geli_swap_flags="-e aes -l 256 -s 4096 -d"

# Монтирование корневой ФС и параметры проверки ФС
root_rw_mount="YES"         # NO = корн. ФС будет монтироваться как read only
fsck_y_enable="NO"          # YES = будет выполнена fsck -y,
                            # если первая попытка проверки неудачна.
fsck_y_flags=""             # Параметры для fsck -y
```

```
background_fsck="YES"       # Если возможно, fsck будет запущен в фоновом режиме
background_fsck_delay="60"  # Время ожидания (в сек.) перед запуском fsck

# Сетевые файловые системы
netfs_types="nfs:NFS nfs4:NFS4 smbfs:SMB portalfs:PORTAL nwfs:NWFS"
extra_netfs_types="NO"

############################################################
###  Сетевые параметры  ####################################
############################################################

### Основные параметры сети и брандмауэра: ###
hostname=""                 # Имя компьютера
hostid_enable="YES"         # UUID хоста
hostid_file="/etc/hostid"   # Файл, содержащий hostuuid
nisdomainname="NO"          # Имя NIS-домена, если используете NIS
dhclient_program="/sbin/dhclient"    # Путь к DHCP-клиенту
dhclient_flags=""           # Параметры dhcp-клиента
#dhclient_flags_fxp0=""     # Параметры dhcp-клиента для интерфейса fxp0
background_dhclient="NO"    # Запускать DHCP-клиент в фоновом режиме
#background_dhclient_fxp0="YES"    # Запускать DHCP-клиент в фоне для fxp0
defaultroute_delay="30"     # Время ожидания для маршрута по умол. от DHCP
# WPA-клиент (беспроводная сеть)
wpa_supplicant_program="/usr/sbin/wpa_supplicant"
wpa_supplicant_flags="-s"   # Параметры wpa_supplicant
wpa_supplicant_conf_file="/etc/wpa_supplicant.conf"

# Брандмауэр
firewall_enable="NO"        # YES = включить брандмауэр
firewall_script="/etc/rc.firewall" # Сценарий конфигурации брамандмауэра
firewall_type="UNKNOWN"     # Тип брандмауэра (см. /etc/rc.firewall)
firewall_quiet="NO"         # YES = не отображать правила
firewall_logging="NO"       # YES = протоколировать события
firewall_flags=""           # Флаги для ipfw

firewall_client_net="192.168.2.0/24" # Адрес сети для клиентского брандмауэра
firewall_simple_iif="ed1"   # Интерфейс для простого брандмауэра (внут. сеть)
firewall_simple_inet="192.168.2.16/28" # Внутренний адрес для простого брандм.
firewall_simple_oif="ed0"   # Внешний интерфейс
firewall_simple_onet="192.0.2.0/28" # Внешний адрес
firewall_myservices=""      # Список TCP-портов
firewall_allowservices=""   # Список IP-адресов, имеющих доступ
                            # к сервисам, "прописанным" в firewall_myservices
firewall_trusted=""         # Список IP-адресов, имеющих полный
                            # доступ к рабочей станции
firewall_logdeny="NO"       # YES = протоколировать запрещенные пакеты
# TCP/UDP порты, для которых не ведется протоколирование
firewall_nologports="135-139,445 1026,1027 1433,1434"
firewall_nat_enable="NO"    # Запретить NAT (если firewall_enable == YES)
firewall_nat_interface=""   # Публичный интерфейс (или IP-адрес) для NAT
firewall_nat_flags=""       # Параметры
dummynet_enable="NO"        # YES = загрузка модуля dummynet
ip_portrange_first="NO"     # Установить первый динамически размещенный порт
ip_portrange_last="NO"      # Установить послед. динамически размещенный порт
ike_enable="NO"             # YES = включить IKE-демон
ike_program="/usr/local/sbin/isakmpd"   # Путь к IKE-демону
```

продолжение ⇩

Листинг 8.4 *(продолжение)*

```
ike_flags=""                # Параметры для IKE-демона
ipsec_enable="NO"           # YES = запуск setkey на ipsec_file
ipsec_file="/etc/ipsec.conf"            # Имя конфиг. файла для setkey

natd_program="/sbin/natd"   # Путь к natd
natd_enable="NO"            # Включить natd (если firewall_enable = YES).
natd_interface=""           # Публичный интерфейс или IP-адрес
natd_flags=""               # Дополнительные флаги

ipfilter_enable="NO"        # YES = включить ipfilter
ipfilter_program="/sbin/ipf"            # Путь к ipfilter
ipfilter_rules="/etc/ipf.rules"         # файл правил для ipfilter
                            # пример: /usr/src/contrib/ipfilter/rules
ipfilter_flags=""           # Параметры ipfilter
ipnat_enable="NO"           # YES = включить ipnat
ipnat_program="/sbin/ipnat" # Путь к ipnat
ipnat_rules="/etc/ipnat.rules"          # файл правил ipnat
ipnat_flags=""              # Параметры ipnat
ipmon_enable="NO"           # YES = вкл. ipmon; нужны ipfilter или ipnat
ipmon_program="/sbin/ipmon" # путь к ipfilter
ipmon_flags="-Ds"           # Параметры ipmon
ipfs_enable="NO"            # YES = сохранить и восстановить состояние
                            # таблиц при завершении работы и загрузке
ipfs_program="/sbin/ipfs"   # Путь к ipfs
ipfs_flags=""               # Параметры ipfs
pf_enable="NO"              # YES = включить фильтр пакетов (pf)
pf_rules="/etc/pf.conf"     # файл правил для pf
pf_program="/sbin/pfctl"    # Путь к pfctl
pf_flags=""                 # Параметры  pfctl
pflog_enable="NO"           # Set to YES to enable packet filter logging
pflog_logfile="/var/log/pflog"          # файл журнала для pflogd
pflog_program="/sbin/pflogd"            # Путь к pflogd
pflog_flags=""              # Параметры pflogd
ftpproxy_enable="NO"        # YES для включения ftp-proxy для pf
ftpproxy_flags=""           # Параметры  ftp-proxy
pfsync_enable="NO"          # Синхронизация состояния pf с другими хостами
pfsync_syncdev=""           # Интерфейс для pfsync
pfsync_syncpeer=""          # IP-адрес pfsync-узла
pfsync_ifconfig=""          # Параметры ifconfig(8) для pfsync
tcp_extensions="YES"        # NO = выключить RFC1323-расширения
log_in_vain="0"             # >=1 протоколировать соединения к портам, на
                            # которых не "висят" демоны
tcp_keepalive="YES"         # Включить тайм-аут для TCP-соединений
tcp_drop_synfin="NO"        # YES = удалить TCP-пакеты с SYN+FIN
icmp_drop_redirect="NO"     # YES = игнорировать пакеты ICMP REDIRECT
icmp_log_redirect="NO"      # YES = протоколировать ICMP REDIRECT пакеты
network_interfaces="auto"   # Список интерфейсов (или "auto").
cloned_interfaces=""        # Список клонированных интерфейсов.
ifconfig_lo0="inet 127.0.0.1"          # Конфигурация обратной петли по умолчанию.
#ifconfig_lo0_alias0="inet 127.0.0.254 netmask 0xffffffff" # Псевдоним.
#ifconfig_ed0_ipx="ipx 0x00010010"     # Запись IPX-адреса.
#ifconfig_em0_name="net0"   # Изменить имя интерфейса с em0 на net0.
#wlans_ath0="wlan0"         # wlan-интерфейс для устройства ath0
#wlandebug_wlan0="scan+auth+assoc"     # Параметры (флаги) для wlanddebug
#ipv4_addrs_fxp0="192.168.0.1/24 192.168.1.1-5/28" # пример записи IPv4-адреса
#
```

```
#autobridge_interfaces="bridge0"       # Список мостов для проверки
#
# Параметры интерфейсов sppp(4)
# См. man spppcontrol(8), чтобы узнать их назначение
sppp_interfaces=""          # Список sppp-интерфейсов
#sppp_interfaces="...0"     # пример: sppp over ...
#spppconfig_...0="authproto=chap myauthname=foo myauthsecret='top secret' hisauthname=some-
gw hisauthsecret='another secret'"
gif_interfaces=""           # Список GIF-интерфейсов
#gif_interfaces="gif0 gif1"

#gifconfig_gif0="10.1.1.1 10.1.2.1"    # Примеры настроек маршрутизатора.
#gifconfig_gif1="10.1.1.2 10.1.2.2"    # Примеры настроек маршрутизатора.
fec_interfaces=""           # Список FEC-интерфейсов Fast EtherChannels
#fec_interfaces="fec0 fec1"
#fecconfig_fec0="fxp0 dc0"             # Если 2 сетевых адаптера
#fecconfig_fec1="em0 em1 bge0 bge1"    # Если 2 сетевых адаптера

# Пользовательские настройки ppp
ppp_enable="NO"             # YES = запустить демон ppp
ppp_program="/usr/sbin/ppp" # Путь к ppp
ppp_mode="auto"             # Выберите "auto", "ddial", "direct" или "dedicated".
                            # Значение по умолчанию: auto.
ppp_nat="YES"               # YES = Использовать NAT PPP
ppp_profile="papchap"       # Используемый профиль из /etc/ppp/ppp.conf
ppp_user="root"             # Пользователь, от имени которого запускается ppp

# Можно запускать сразу несколько копий PPP при загрузке
#ppp_profile="profile1 profile2 profile3"    # Используемые профили
#ppp_profile1_mode="ddial"  # Переопределить режим PPP для профиля 1
#ppp_profile2_nat="NO"      # Переопределить режим NAT для профиля 1

### Загрузка сетевых демонов ###
hostapd_enable="NO"         # YES = запуск демон hostap
syslogd_enable="YES"        # YES = запуск демон syslog (или NO).
syslogd_program="/usr/sbin/syslogd" # Путь к syslogd
syslogd_flags="-s"          # Параметры syslogd (
inetd_enable="NO"           # Суперсервер inetd
inetd_program="/usr/sbin/inetd"    # Путь к inetd
inetd_flags="-wW -C 60"     # Параметры inetd
#
# Сервер DNS named.
#
named_enable="NO"           # YES = запуск сервера DNS
named_program="/usr/sbin/named"             # Путь к named, если отличается по умолчанию
named_conf="/etc/namedb/named.conf"         # Путь к конфиг. файлу
#named_flags="-c /etc/namedb/named.conf"    # Расскомментируйте, если named
                                            # не в /usr/sbin
named_pidfile="/var/run/named/pid"          # PID-файл DNS-сервера

# Пользователь, от имени которого запускается named
named_uid="bind"
named_chrootdir="/var/named"            # Chroot-каталог
named_chroot_autoupdate="YES"           # Автообновление chroot-окружения

named_symlink_enable="YES"  # Ссылка на PID-файл в chroot-окружении
```

продолжение ⇩

Листинг 8.4 *(продолжение)*

```
named_wait="NO"             # Ожидать ответа named перед выходом
named_wait_host="localhost" # Имя узла для проверки, если named_wait включен
named_auto_forward="NO"     # Настраивать NDS-серверы в /etc/resolv.conf
named_auto_forward_only="NO"     # "forward only" вместо "forward first"

#
# Настройки протокола kerberos
#
kerberos5_server_enable="NO"            # YES = запустить мастер-сервер kerberos 5
kerberos5_server="/usr/libexec/kdc"     # Путь к kerberos 5 KDC
kerberos5_server_flags="--detach"       # Доп. параметры сервера kerberos
kadmind5_server_enable="NO"             # YES = запустить kadmind
kadmind5_server="/usr/libexec/kadmind"  # Путь к демону kerberos 5 (admin)
kpasswdd_server_enable="NO"             # YES = запустить kpasswdd
kpasswdd_server="/usr/libexec/kpasswdd" # Путь к демону kerberos 5 (passwd)

gssd_enable="NO"            # YES = запустить демон gssd
gssd_flags=""               # Параметры gssd.

rwhod_enable="NO"           # YES = запустить демон rwho
rwhod_flags=""              # Параметры rwhod
rarpd_enable="NO"           # YES = запустить rarpd
rarpd_flags="-a"            # Параметры rarpd.
bootparamd_enable="NO"      # YES = запустить bootparamd
bootparamd_flags=""         # Параметры bootparamd
pppoed_enable="NO"          # YES = запустить демон pppoed
pppoed_provider="*"
pppoed_flags="-P /var/run/pppoed.pid"   # Параметры pppoed
pppoed_interface="fxp0"                 # Интерфейс pppoed

pppoed sshd_enable="NO"                 # YES = запустить sshd
sshd_program="/usr/sbin/sshd"           # Путь к sshd, если нужно указать другой.
sshd_flags=""                           # Параметры sshd.
ftpd_enable="NO"                        # YES = включить автономный ftpd
ftpd_program="/usr/libexec/ftpd"        # Путь к ftpd
ftpd_flags=""                           # Параметры автономного ftpd

### Сетевой демон (NFS): для всех нужно rpcbind_enable="YES" ###

amd_enable="NO"                         # YES = запуск сервера amd с флагами $amd_flags.
amd_program="/usr/sbin/amd"             # Путь к amd, если нужно указать другой.
amd_flags="-a /.amd_mnt -l syslog /host /etc/amd.map /net /etc/amd.map"
amd_map_program="NO"                    #
nfs_client_enable="NO"                  # Узел для NFS-клиента(или NO).
nfs_access_cache="60"                   # Тайм-аут клиентского кэша в секундах
nfs_server_enable="NO"                  # YES = включить NFS-сервер
nfs_server_flags="-u -t -n 4"           # Параметры nfsd
mountd_enable="NO"                      # YES = запустить mountd
mountd_flags="-r"                       # Параметры mountd
weak_mountd_authentication="NO"         # Разрешить запросы
                                        # монтирования от не-root пользователей
nfs_reserved_port_only="NO"             # YES = предоставить NFS безопасный порт
nfs_bufpackets=""                       # Размер буфера для клиента (в пакетах)
rpc_lockd_enable="NO"                   # Запустить блокировку, необх. для клиент/сервера
```

```
rpc_lockd_flags=""                      # Параметры rpc.lockd (если включен).
rpc_statd_enable="NO"                   # Запустить NFS rpc., необх. для клиент/сервера
rpc_statd_flags=""                      # Параметры rpc.statd (если включен).
rpcbind_enable="NO"                     # YES = запустить portmapper service
rpcbind_program="/usr/sbin/rpcbind"     # Путь к rpcbind
rpcbind_flags=""                        # Параметры rpcbind (если вкл.)
rpc_ypupdated_enable="NO"               # YES = запустить NIS-мастер или SecureRPC
keyserv_enable="NO"                     # YES = запустить сервер ключей SecureRPC
keyserv_flags=""                        # Параметры keyserv (если вкл.).
nfsv4_server_enable="NO"                # YES = включить поддержку NFSv4
nfscbd_enable="NO"                      # NFSv4 клиентский демон обратной связи
nfscbd_flags=""                         # Параметры nfscbd
nfsuserd_enable="NO"                    # NFSv4-демон маппинга имени пользователя/пароля
nfsuserd_flags=""                       # Параметры nfsuserd

### Сервисы времени: ###
timed_enable="NO"                       # YES = запустить демон времени timed
timed_flags=""                          # Параметры timed (если вкл.)
# YES = Запустить ntpdate для синх. времени при загрузке:
ntpdate_enable="NO"
ntpdate_program="/usr/sbin/ntpdate"     # Путь к ntpdate
ntpdate_flags="-b"                      # Параметры ntpdate (если вкл.)
ntpdate_config="/etc/ntp.conf"          # Конфиг. файл ntpdate
ntpdate_hosts=""                        # Список серверов ntpdate, разделенный пробелами
ntpd_enable="NO"                        # YES = запустить ntpd, Network Time Protocol
ntpd_program="/usr/sbin/ntpd"           # Путь к ntpd
ntpd_config="/etc/ntp.conf"             # Конфиг. файл ntpd
ntpd_sync_on_start="NO"                 # YES = синхронизация времени при запуске
ntpd_flags="-p /var/Запустить/ntpd.pid -f /var/db/ntpd.drift"
                                        # Параметры ntpd (если вкл.)

##################################
# Параметры Network Information Services (NIS):
# Все требуют rpcbind_enable="YES" #
##################################
nis_client_enable="NO"                  # Мы - NIS-клиент (NO = мы не NIS-клиент)
nis_client_flags=""                     # Параметры ypbind (если вкл.)
nis_ypset_enable="NO"                   # YES = запустить ypset при загрузке
nis_ypset_flags=""                      # Параметры ypset (если вкл.)
nis_server_enable="NO"                  # Мы - NIS-сервер?
nis_server_flags=""                     # Параметры ypserv (если вкл.)
nis_ypxfrd_enable="NO"                  # YES = запустить rpc.ypxfrd во время загрузки
nis_ypxfrd_flags=""                     # Параметры rpc.ypxfrd (если вкл.)
nis_yppasswdd_enable="NO"               # YES = запустить rpc.yppasswdd во время
nis_yppasswdd_flags=""                  # Параметры rpc.yppasswdd (если вкл.)

### SNMP-демон ###
bsnmpd_enable="NO"                      # YES = запустить SNMP-демон (или NO).
bsnmpd_flags=""                         # Параметры bsnmpd.

### Параметры маршрутизации: ###
defaultrouter="NO"                      # Шлюз по умолчанию (или NO)
static_routes=""                        # Список статичных маршрутов
natm_static_routes=""                   # Список статичных маршрутов для NATM
gateway_enable="NO"                     # YES, если компьютер - шлюз
router_enable="NO"                      # YES = запустить демон маршрутизации
```

продолжение ⇩

Листинг 8.4 *(продолжение)*

```
router="/sbin/routed"                   # Имя демона маршрутизации
router_flags="-q"                       # Параметры демона маршрутизации
mrouted_enable="NO"                     # Включить широковещательную машрутизацию IPv4
mrouted_program="/usr/local/sbin/mrouted"    # Имя демона маршрутизации IPv4
                                        # mrouted нужно устанавливать из портов
mrouted_flags=""                        # Параметры демона mrouted
ipxgateway_enable="NO"                  # YES для включения IPX-маршрутизации (шлюз)
ipxrouted_enable="NO"                   # YES для запуска IPX-демона (маршрутизатор)
ipxrouted_flags=""                      # Параметры демона IPX-маршрутизации
arpproxy_all="NO"
forward_sourceroute="NO"                # маршрутизация источника
                                        # (только если gateway_enable установлен в "YES")
accept_sourceroute="NO"                 # принимать пакеты маршрутизации источника

### Опции ATM-интерфейса: ###
atm_enable="NO"                         # YES = включить поддержку ATM-интерфейсов
#atm_netif_hea0="atm 1"                 # Сетевой адрес физического интерфейса
#atm_sigmgr_hea0="uni31"                # Сигнальный менеджер для физического интерфейса
#atm_prefix_hea0="ILMI"                 # Префикс NSAP
#atm_macaddr_hea0="NO"                  # Перезаписать физический MAC-адрес
# Адрес ATMARP-сервера (или local).
#atm_arpserver_atm0="0x47.0005.80.999999.9999.9999.9999.999999999999.00"
#atm_scsparp_atm0="NO"                  # Запустить SCSP/ATMARP на сетев. интерфейсе
atm_pvcs=""                             # PVC-список
atm_arps=""                             # Постоянный ARP-список

### Поддержка Bluetooth ###
hcsecd_enable="NO"                      # YES = включить hcsecd
hcsecd_config="/etc/bluetooth/hcsecd.conf" # Файл конфигурации hcsecd

sdpd_enable="NO"                        # YES = включить sdpd
sdpd_control="/var/run/sdp"             # Контрольный сокет sdpd
# Имя пользователя и группы, от имени которых запускается sdpd
sdpd_groupname="nobody"
sdpd_username="nobody"

bthidd_enable="NO"                      # YES = включить bthidd
bthidd_config="/etc/bluetooth/bthidd.conf" # Конфигурационный файл bthidd
bthidd_hids="/var/db/bthidd.hids"       # Файл известных HID-устройств

rfcomm_pppd_server_enable="NO"          # YES = rfcomm_pppd в режиме сервера
rfcomm_pppd_server_profile="one two"    # Профиль из /etc/ppp/ppp.conf
#

### Разные сетевые опции: ###

# Разрешено отвечать на широковещательные пакеты ping
icmp_bmcastecho="NO"

### Опции IPv6: ###
ipv6_enable="NO"                        # YES = поддержка IPv6
ipv6_network_interfaces="auto"          # Список сетевых интерфейсов
ipv6_defaultrouter="NO"                 # Шлюз по умолчанию IPv6
#ipv6_defaultrouter="2002:c058:6301::"  # Шлюз по умолчанию (IPv6)
```

```
ipv6_static_routes=""                   # Список статических маршрутов

#ipv6_route_xxx="fec0:0000:0000:0006:: -prefixlen 64 ::1"
ipv6_gateway_enable="NO"                # YES, если компьютер - шлюз
ipv6_router_enable="NO"                 # YES = включить IPv6-маршрутизацию
ipv6_router="/usr/sbin/route6d"         # Имя демона маршрутизации IPv6
ipv6_router_flags=""                    # Параметры демона маршрутизации IPv6
#ipv6_network_interfaces="ed0 ep0"      # Пример для маршрутизатора

#ipv6_prefix_ed0="fec0:0000:0000:0001 fec0:0000:0000:0002"
#ipv6_prefix_ep0="fec0:0000:0000:0003 fec0:0000:0000:0004"
ipv6_default_interface="NO"             # Интерфейс по умолчанию или NO, если не задан

rtsol_flags=""                          # Параметры IPv6-маршрутизатора
rtadvd_enable="NO"                      # YES = включить IPv6-маршрутизацию

rtadvd_interfaces=""                    # Интерфейсы rtadvd
mroute6d_enable="NO"                    # Широковещательная IPv6 маршрутизация
mroute6d_program="/usr/local/sbin/pim6dd"    # Имя демона маршрутизации,

mroute6d_flags=""                       # Параметры демона маршрутизации IPv6
stf_interface_ipv4addr=""               # Локальный IPv4-адрес для туннельного интерфейса

stf_interface_ipv4plen="0"              # Длина префикса для 6to4 IPv4 адреса

stf_interface_ipv6_ifid="0:0:0:1"       # IPv6 интерфейс для stf0.
                                        # Можно установить "AUTO"
stf_interface_ipv6_slaid="0000"
ipv6_faith_prefix="NO"

ipv6_ipv4mapping="NO"                   # YES = включить маппинг адресов IPv4 - IPv6
ipv6_firewall_enable="NO"               # YES = включить IPv6-брандмауэра

# конфигурация брандмауэра для IPv6
ipv6_firewall_script="/etc/rc.firewall6"
ipv6_firewall_type="UNKNOWN"            # Тип брандмауэра IPv6 (см. /etc/rc.firewall6)
ipv6_firewall_quiet="NO"                # YES = не выводить правила
ipv6_firewall_logging="NO"              # YES = протоколировать события
ipv6_firewall_flags=""                  # Параметры ip6fw
ipv6_ipfilter_rules="/etc/ipf6.rules"   # файл правил для ipfilter,
                                        # см. /usr/src/contrib/ipfilter/rules

ip6addrctl_enable="YES"                 # YES = выбор адреса по умолчанию
ip6addrctl_verbose="NO"                 # YES = вывод подробных сообщений

###########################################################
###  Опции системной консоли  #############################
###########################################################

keyboard=""                     # Устройство клавиатуры (по умолчанию /dev/kbd0).
keymap="NO"                     # Раскладка в /usr/share/syscons/keymaps/* (или NO).
keyrate="NO"                    # скорость клавиатуры: slow, normal, fast (или NO).
keybell="NO"                    # См. man kbdcontrol(1).
keychange="NO"                  # Значения по умолчанию для функ. клавиш (или NO).
cursor="NO"                     # Тип курсора {normal|blink|destructive} (или NO).
scrnmap="NO"                    # Раскладка экрана из /usr/share/syscons/scrnmaps/
```

продолжение ⇩

Листинг 8.4 *(продолжение)*

```
font8x16="NO"                   # Шрифт 8x16 из /usr/share/syscons/fonts/* (или NO).
font8x14="NO"                   # Шрифт 8x14 из /usr/share/syscons/fonts/* (или NO).
font8x8="NO"                    # Шрифт 8x8 из /usr/share/syscons/fonts/* (или NO).
saver="NO"                      # Скринсейвер: использует /boot/kernel/${saver}_saver.ko
moused_enable="NO"              # YES = включить демон мыши.
moused_type="auto"              # См. man rc.conf
moused_port="/dev/psm0"         # Порт мыши
moused_flags=""                 # Параметры moused.
mousechar_start="NO"
allscreens_flags=""             # Использовать режим vidcontrol для всех вирт. консолей
allscreens_kbdflags=""          # Использовать режим kbdcontrol для всех вирт. консолей

###########################################################
###  Параметры агента передачи почты (MTA)  ###############
###########################################################

# Сценарий для запуска используемого MTA, по умолчанию sendmail
mta_start_script="/etc/rc.sendmail"

# Параметры для /etc/rc.sendmail и /etc/rc.d/sendmail:
sendmail_enable="NO"                            # YES = запустить sendmail
sendmail_pidfile="/var/Запустить/sendmail.pid"  # PID-файл sendmail
sendmail_procname="/usr/sbin/sendmail"          # имя процесса sendmail
sendmail_flags="-L sm-mta -bd -q30m"            # Параметры sendmail
sendmail_submit_enable="YES"
sendmail_submit_flags="-L sm-mta -bd -q30m -ОДемонPortOptions=Addr=localhost"

# Параметры только локального MTA
sendmail_outbound_enable="YES"
sendmail_outbound_flags="-L sm-queue -q30m"     # Параметры sendmail
sendmail_msp_queue_enable="YES"
# Параметры демона sendmail_msp_queue
sendmail_msp_queue_flags="-L sm-msp-queue -Ac -q30m"

sendmail_rebuild_aliases="NO"                   # YES = перестроить псевдонимы


###########################################################
###  Различные параметры ##################################
###########################################################

auditd_enable="NO"              # YES = включить демон audit.
auditd_program="/usr/sbin/auditd" # Путь к демону audit.
auditd_flags=""                 # Параметры демона аудита.

cron_enable="YES"               # YES = включить cron
cron_program="/usr/sbin/cron"   # путь к cron (если вкл.).
cron_dst="YES"
cron_flags=""                   # Параметры cron.
lpd_enable="NO"                 # YES = включить демон поддержки печати
lpd_program="/usr/sbin/lpd"     # Путь к lpd
lpd_flags=""                    # Параметры lpd

nscd_enable="NO"                # YES = включить кэширующий демон nsswitch
```

```
chkprintcap_enable="NO"          # YES = запустить chkprintcap перед lpd
chkprintcap_flags="-d"           # Создать отсутствующие каталоги.
dumpdev="NO"                     # Устройство для дампа
dumpdir="/var/crash"             # Каталог для дампа
savecore_flags=""                # Параметры сохранения дампа
crashinfo_enable="YES"           # YES = автоотчет о сбое
crashinfo_program="/usr/sbin/crashinfo"   # Программа для создания отчета

quota_enable="NO"                # YES = включить квоты
check_quotas="YES"               # YES = проверять квоты при запуске
quotaon_flags="-a"               # Вкл. квоты для всех файловых систем
quotaoff_flags="-a"              # Выкл. квоты для всех ФС при завершении работы
quotacheck_flags="-a"            # Проверить квоты всех ФС (если вкл.)
accounting_enable="NO"           # YES = включить учет процессов

# Службы эмуляции/совместимости из /etc/rc.d/abi
sysvipc_enable="NO"              # YES = загрузить примитивы System V IPC при запуске
linux_enable="NO"                # YES = совместимость с бинарными файлами Linux
svr4_enable="NO"                 # YES = эмуляция SysVR4 при загрузке
clear_tmp_enable="NO"            # YES = очистить /tmp при загрузке
clear_tmp_X="YES"                # YES = очистить временные файлы X при запуске
ldconfig_insecure="NO"           # YES = выкл. проверку безопасности ldconfig

ldconfig_paths="/usr/lib/compat /usr/local/lib /usr/local/lib/compat/pkg"
ldconfig32_paths="/usr/lib32"                    # 32-битные библиотеки
ldconfig_paths_aout="/usr/lib/compat/aout /usr/local/lib/aout"

# Локальные каталоги с конф. файлами ldconfig
ldconfig_local_dirs="/usr/local/libdata/ldconfig"
ldconfig_local32_dirs="/usr/local/libdata/ldconfig32"
            # Локальные каталоги с конф. файлами ldconfig
            # 32-битная совместимость.
kern_securelevel_enable="NO"     # уровень безопасности ядра (см. man security(7))
kern_securelevel="-1"            # диапазон: -1..3 ; '-1' самый небезопасный

update_motd="YES"                # YES = обновить версию в /etc/motd
entropy_file="/entropy"          # NO для отключения кэширования
                                 # энтропии при перезагрузке

entropy_dir="/var/db/entropy"    # Каталог энтропии
entropy_save_sz="2048"           # Размер кэширующих файлов энтропии.
entropy_save_num="8"             # Количество кэширующих файлов энтропии

dmesg_enable="YES"               # Сохранить вывод dmesg /var/run/dmesg.boot
watchdogd_enable="NO"            # YES = запустить демон watchdog

# Файлы с правилами devfs
devfs_rulesets="/etc/defaults/devfs.rules /etc/devfs.rules"

performance_cx_lowest="HIGH"     # Онлайн-состояние простоя CPU
performance_cpu_freq="NONE"      # Онлайн-частота CPU
economy_cx_lowest="HIGH"         # Оффлайн-состояние простоя CPU
economy_cpu_freq="NONE"          # Оффлайн-частота CPU
newsyslog_enable="YES"           # YES = включить newsyslog
newsyslog_flags="-CN"            # Параметры Newsyslog
mixer_enable="YES"               # YES = запустить звуковой миксер.
```

продолжение ⇩

Листинг 8.4 *(продолжение)*

```
############################################################
### Параметры chroot-окружения ###########################
############################################################
jail_enable="NO"                # NO = включить chroot-окружения
jail_list=""                    # Список имен окружений, разделенных пробелами

# Разрешить пользователя root в chroot, чтобы изменять имя узла
jail_set_hostname_allow="YES"
jail_socket_unixiproute_only="YES"          # Маршрут TCP/IP в chroot-окружении
jail_sysvipc_allow="NO"                     # Разрешить SystemV IPC в chroot-окружении

#jail_example_rootdir="/usr/jail/default"    # корневой каталог chroot-окружения

# имя машины в chroot-окружении
#jail_example_hostname="default.domain.com"
#jail_example_interface=""                  # интерфейс chroot-окружения
#jail_example_fib="0"                       # Таблица маршрутизации
#jail_example_ip="192.0.2.10,2001:db8::17"  # IPv4/IPv6-адреса
#jail_example_ip_multi0="2001:db8::10"      # второй IPv6-адрес

# команда для выполнения в chroot-окружении при запуске окружения
#jail_example_exec_start="/bin/sh /etc/rc"
# команда, которая будет выполнена после запуска chroot-окружения
#jail_example_exec_afterstart0="/bin/sh command"
# команда останова chroot-окружения
#jail_example_exec_stop="/bin/sh /etc/rc.shutdown"
#jail_example_devfs_enable="NO"             # YES = монтировать devfs
#jail_example_devfs_ruleset="ruleset_name"  # правила devfs
#jail_example_fdescfs_enable="NO"           # YES = монтировать fdescfs в окружении
#jail_example_procfs_enable="NO"            # YES = монтировать procfs в окружении
#jail_example_mount_enable="NO"             # mount/umount ФС chroot-окр.
#jail_example_fstab=""                      # fstab для mount/umount
#jail_example_flags="-l -U root"            # Параметры jail

############################################################
```

Далее следует bash-код, который нам не интересен. Можете просмотреть этот файл, чтобы узнать, чем он заканчивается: cat /etc/defaults/r.conf

Управление учетными записями пользователей

9

9.1. Однопользовательские и многопользовательские системы

Вспомним DOS. Наверное, нет такого пользователя в наше время, который бы не видел DOS. Уж если дома и был современный компьютер, то в некоторых учебных заведениях до сих пор установлена эта операционная система[1].

DOS — это однозадачная и однопользовательская операционная система. Это означает, что в один и тот же момент времени с системой может работать только один пользователь, а процессор может быть занят только одной программой. Конечно, вы можете возразить, что, начиная с пятой версии DOS, в ее ядре появился «зародыш» многозадачности — именно в этой версии появилась возможность запускать резидентные программы, но все равно назвать DOS многозадачной язык не поворачивается.

UNIX (а значит, и FreeBSD, и Linux) — многозадачная и многопользовательская система. В один момент времени может быть запущено несколько процессов и с системой могут работать несколько пользователей (конечно, не за одним монитором, а по сети).

Каждый пользователей работает в системе под своей учетной записью. При установке была создана ваша учетная запись, но вы можете создать учетные записи для всех, кто будет работать с вашим компьютером. Это очень удобно. Ведь у каждого пользователя могут быть свои настройки программ. К тому же никто не знает, как вам в следующий раз захочется войти в систему, а способы входа в систему могут быть разными: возможна регистрация в консоли, регистрация через X-сервер (графический режим), удаленная регистрация по SSH, FTP, а также удаленная регистрация в графическом режиме (удаленный рабочий стол).

---

[1] Уже и такие пользователи встречаются. И в немалом количестве, и даже среди потенциальных читателей данной книги.

9.2. Особенности управления пользователями в Linux

В Linux есть очень удобные графические конфигураторы, позволяющие добавить/удалить пользователя или группу пользователей, модифицировать учетную запись пользователя. Но использовать их не интересно — все очень просто, и с добавлением пользователя справится даже ребенок. Я обещал, что в этой книге мы, по возможности, будем стараться меньше использовать конфигураторы, зато будем все делать средствами самой системы. И так даже лучше — когда вы установите другой дистрибутив Linux, вам не нужно будет искать, как называется конфигуратор управления пользователями. Команды `adduser` (в некоторых дистрибутивах — `useradd`) и `passwd` есть в любом дистрибутиве.

9.2.1. Добавление пользователей

Сначала добавим нового пользователя. Для этого используется команда `adduser`. Данная команда только добавляет учетную запись пользователя в файл /etc/passwd (листинг 9.1), но не устанавливает его пароль. Для установки пароля используется другая команда — `passwd`, которая принимает введенный администратором пароль и записывает его в файл /etc/shadow. Поэтому для добавления пользователя нужно ввести две команды:

```
# adduser имя_пользователя
# passwd имя_пользователя
```

Листинг 9.1. Пример файла /etc/passwd

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/bin/sh
daemon:x:2:2:daemon:/sbin:/bin/sh
adm:x:3:4:adm:/var/adm:/bin/sh
lp:x:4:7:lp:/var/spool/lpd:/bin/sh
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/bin/sh
news:x:9:13:news:/var/spool/news:/bin/sh
uucp:x:10:14:uucp:/var/spool/uucp:/bin/sh
operator:x:11:0:operator:/var:/bin/sh
games:x:12:100:games:/usr/games:/bin/sh
nobody:x:65534:65534:Nobody:/:/bin/sh
rpm:x:13:101:system user for rpm:/var/lib/rpm:/bin/false
messagebus:x:14:105:system user for dbus:/:/sbin/nologin
avahi:x:15:106:system user for avahi:/var/avahi:/bin/false
haldaemon:x:16:16:system user for hal:/:/sbin/nologin
vcsa:x:69:69:virtual console memory owner:/dev:/sbin/nologin
gdm:x:70:70:system user for gdm:/var/lib/gdm:/bin/false
den:x:500:500:den:/home/den:/bin/bash
```

Как видно из листинга 9.1, формат /etc/passwd следующий:

```
Имя_пользователя:пароль:UID:GID:комментарий:домашний_каталог:оболочка
```

UID — уникальный идентификатор пользователя, а GID — группы. Поскольку пароли хранятся в файле /etc/shadow, во втором поле всегда будет «x». В файле /etc/shadow есть восемь полей:

- Имя пользователя — совпадает с именем пользователя из /etc/passwd.
- Пароль — зашифрованный пароль, обычно это поле начинается с 1, что свидетельствует об использовании алгоритма шифрования MD5.
- Дата последнего изменения пароля — содержит количество дней, прошедших с 1 января 1970 года до даты последнего изменения пароля.
- Минимум — минимальное число дней между изменениями пароля, то есть число дней, которые должны пройти с даты последнего изменения пароля до следующего изменения пароля.
- Максимум — по прошествии этого количества дней учетная запись будет заблокирована для принудительного изменения пароля.
- Предупреждение — число дней, по истечению которых пользователь получит предупреждение, что скоро придется сменить пароль.
- Блокировка — число дней, которое должно пройти после истечения срока действия пароля (поле Максимум) до отключения учетной записи.
- Дата окончательной блокировки — абсолютная дата, то есть количество дней с 1 января 1970 года до окончательной блокировки учетной записи. Это поле удобно использовать для «временных» сотрудников — в это поле можно внести дату окончания трудового договора (контракта).

Команду passwd может использовать не только администратор, но и сам пользователь — для изменения собственного пароля, при этом ему не нужно указывать имя пользователя, а просто ввести команду passwd.

9.2.2. Модификация и удаление пользователей

Для удаления пользователя используется команда userdel (или deluser в некоторых дистрибутивах):

```
# userdel <имя_пользователя>
```

Для модификации пользователя используется команда usermod, о ее использовании вы можете прочитать в справочной системе (man usermod). Однако Linux — это не FreeBSD, и вы можете использовать любой текстовый редактор для редактирования /etc/password вручную без использования usermod, однако с помощью usermod удобнее указывать даты истечения пароля, блокирования учетной записи т. д.

9.3. Особенности управления пользователями во FreeBSD

9.3.1. Добавление пользователей

Во FreeBSD для добавления пользователей тоже используется команда adduser, но используется она иначе, в интерактивном режиме. Вы запускаете команду

adduser, никаких параметров передавать ей не нужно, затем adduser задаст вам ряд вопросов:

```
# adduser
Username: ivan
Full name: Ivan Ivanov
Uid (Leave empty for default):
Login group [ivan]:
Login group is ivan. Invite max into other groups? []:
Login class [default]: russian
Shell (sh csh tcsh bash rbash nologin) [sh]:
Home directory [/home/ivan]:
Use password-based authentication? [yes]: yes
Use an empty password? (yes/no) [no]: no
Use a random password? (yes/no) [yes]: no
Enter password:
Enter password again:
Lock out the account after creation? [no]: no
Username : ivan
Password : ****
Full Name : Ivan Ivanov
Uid : 1004
Class :
Groups : ivan
Home : /home/ivan)
Shell : /bin/sh
Locked : no
OK? (yes/no): yes
adduser: INFO: Successfully added (ivan) to the user database.
Add another user? (yes/no): no
```

Сначала нужно ввести имя пользователя, которое будет использоваться для входа в систему. Постарайтесь придумать не очень длинный логин, чтобы пользователю было удобно его вводить. Далее нужно ввести полное имя пользователя. Эта информация важна больше для вас, чем для самого пользователя — вы же не можете запомнить фамилии всех пользователей вашего предприятия. Зато потом точно сможете ответить начальству, кто именно посещал сайты для взрослых. UID (идентификатор пользователя) указывать не нужно — система присвоит его автоматически, используя следующий свободный идентификатор. Далее создается логин-группа, как правило, ее название совпадает с именем пользователя, поэтому вводить ее название не нужно.

Затем вы можете добавить пользователя в другие группы, если считаете нужным. Следующий вопрос — это класс регистрации; чтобы консоль у пользователя была на русском языке, укажите russian. После класса регистрации нужно указать оболочку пользователя (в скобках выводятся допустимые имена), по умолчанию используется оболочка sh.

После ввода оболочки нужно ответить yes или no на несколько вопросов:

- Использовать аутентификацию на основе паролей? Как правило, нужно ответить yes или просто нажать Enter.
- Использовать пустой пароль? Этого делать не нужно, создайте для пользователя пароль, хотя бы самый простой.

- Использовать случайный пароль? Здесь решайте сами: «случайные» пароли сложнее для подбора, но и сложнее для запоминания.
- Если вы на предыдущий вопрос ответили no, то нужно ввести пароль и повторить ввод.

Затем программа предложит просмотреть информацию о созданном пользователе:

```
Full Name : Ivan Ivanov
Uid : 1004
Class :
Groups : ivan
Home : /home/ivan)
Shell : /bin/sh
Locked : no
OK? (yes/no): yes
```

Если все нормально, введите yes. Вы получите сообщение о том, что пользователь добавлен в базу данных, и программа предложит создать еще одного пользователя.

При добавлении пользователя соответствующая запись добавляется в файл /etc/passwd. Формат этого файла такой же, как и в Linux. А вот зашифрованный пароль помещается в файле /etc/master.passwd, а не в /etc/shadow. Однако во FreeBSD кроме файлов /etc/passwd и /etc/master.passwd используются еще две хеш-таблицы, хранящиеся в файлах /etc/pwd.db и /etc/spwd.db. Именно поэтому редактировать /etc/passwd (например, если вы просто хотите изменить фамилию пользователя или его оболочку) в любом текстовом редакторе нельзя, нужно обязательно использовать программу vipw. Программа vipw запустит текстовый редактор для редактирования именно таблицы пользователей, которая строится путем объединения полей файлов /etc/passwd и /etc/master.passwd. Формат файла /etc/master.passwd почти такой же, как и у /etc/passwd, но после поля GID добавляются три поля: класс регистрации, время смены пароля, время блокировки учетной записи. Время задается в секундах, прошедших с 1 января 1970 года до заданной даты (до даты смены пароля или блокировки учетной записи). При сохранении такой объединенной таблицы вносятся изменения в файлы /etc/passwd и /etc/master.passwd, а после этого на основании этих файлов строятся хеш-таблицы /etc/pwd.db и /etc/spwd.db. Если вы вручную (с помощью любого текстового редактора) изменили файлы /etc/passwd и /etc/master.passwd, то введите команду pwd mkdb.

Перед запуском программы vipw настоятельно рекомендую в переменной окружения EDITOR указать дружественный редактор (лучше всего использовать редактор ee), иначе будет вызван классический редактор vi, который своей «функциональностью» удручает любого современного пользователя.

9.3.2. Удаление пользователя

Для удаления пользователя нужно использовать команду rmuser, в качестве аргумента указать имя удаляемого пользователя. Программа спросит вас, желаете ли вы также удалить и домашний каталог пользователя.

```
# rmuser ivan
Matching password entry:
ivan:*:1004:1004::0:0:Ivan Ivanov:/home/ivan:/bin/sh
Is this the entry you wish to remove? y
Remove user's home directory (/home/ivan)? y
Removing user (ivan): mailspool home passwd.
```

9.3.3. Изменение пароля

Для изменения пароля пользователя, как и в Linux, используется команда `passwd`, которая может изменить пароль текущего пользователя, а также любого пользователя, указанного в качестве параметра, при условии, что команду `passwd` вызывает root или пользователь с правами root.

9.4. Группы пользователей

Сведения о группах пользователей хранятся в файле /etc/group (листинг 9.2). Для добавления, изменения и удаления групп пользователей используются соответственно,команды `groupadd`, `groupmod`, `groupdel`. Прочитать об использовании данных команд можно в справочной системе: `man <имя_команды>`.

Листинг 9.2. Пример файла /etc/group

```
root:x:0:
bin:x:1:
daemon:x:2:messagebus,haldaemon
sys:x:3:
adm:x:4:
tty:x:5:
disk:x:6:
lp:x:7:
mem:x:8:
kmem:x:9:
wheel:x:10:
mail:x:12:
news:x:13:
uucp:x:14:
man:x:15:
floppy:x:19:
games:x:20:
tape:x:21:
cdrom:x:22:
utmp:x:24:
usb:x:43:
cdwriter:x:80:
audio:x:81:
video:x:82:
users:x:100:
nogroup:x:65534:
rpm:x:101:
xgrp:x:102:gdm
```

```
ntools:x:103:
ctools:x:104:
messagebus:x:105:
avahi:x:106:
haldaemon:x:16:
vcsa:x:69:
gdm:x:70:
slocate:x:107:
den:x:500:
```

Формат файла /etc/group следующий:

```
Имя_группы:пароль:GID:список
```

Последнее поле содержит список пользователей, относящихся к группе, разделенный запятыми.

9.5. Пользователь root

Пользователь root обладает максимальными правами в системе, поэтому нужно стараться как можно меньше работать под именем root. Почему? Чтобы уберечь систему от самого себя. Можно нечаянно ввести разрушительную команду, и система выполнит ее, не задавая лишних вопросов в стиле Windows: «А вы уверены?» К тому же к вашей консоли в обеденный перерыв может пробраться недоброжелатель (когда кушать хочется, о команде `exit` забываешь), что он может сотворить — одному ему известно.

Исходя из всего этого, для повседневной работы нужно создать обычного пользователя. Под этим пользователем вы можете просматривать большинство конфигурационных файлов, читать журналы системы, но не можете навредить системе. Когда же понадобятся права root, можно или войти как root, или использовать команду `su`. Когда вы введете команду `su`, увидите предложение ввести пароль root. Если введенный пароль правильный, вы получите доступ к консоли root. По окончании работы в привилегированном режиме не забудьте ввести команду `exit`, чтобы завершить root-сеанс.

У вас, как у любого нормального и занятого администратора, могут быть помощники — младшие администраторы. Хотя бы один помощник должен быть, поскольку «сфера информационных технологий» всегда требует присмотра, а вы можете заболеть, уехать в отпуск и т. д. Не оставлять же сервер без присмотра? Но и сообщать пароль root младшему администратору тоже не хочется. Мало того, что придется делиться своим паролем, что не очень хорошо, поскольку он может использоваться также для доступа к вашему почтовому ящику, входа на страницу в социальной сети и т. д. Не секрет, что многие пользователи используют один и тот же пароль для доступа к разным ресурсам. К тому же не хочется младшему администратору оставлять полный контроль над сервером, хочется как-то ограничить его права, хотя бы определить, какие команды он может вводить, а какие — нет. Все это позволяет организовать команда `sudo`. Давайте разберемся, как ее правильно настроить и использовать.

В Linux программа `sudo` установлена по умолчанию. А вот во FreeBSD нужно ее установить самостоятельно.

```
# cd /usr/ports/security/sudo/
# make install clean
# rehash
```

После установки программа нуждается в настройке. В Linux ее тоже нужно настроить, поскольку в некоторых дистрибутивах sudo имеет право по умолчанию использовать каждый зарегистрированный пользователь системы, что не есть хорошо — не может же быть администратором каждый пользователь?

Нужно определить список пользователей, которые могут использовать sudo. Этот список в Linux хранится в файле /etc/sudoers, а во FreeBSD — в /usr/local/etc/sudoers. Но редактировать этот файл напрямую нельзя, нужно использовать команду visudo. Предположим, что логин нашего младшего админа — admin. Если вы желаете разрешить ему выполнять все команды даже без ввода пароля, тогда добавьте следующую строку в список sudoers:

```
admin ALL=(ALL) NOPASSWD: ALL
```

Этой строчкой вы фактически предоставляете пользователю admin максимальные права. Он сможет выполнить любую команду, но через команду sudo, например:

```
sudo rm -rf /
```

Вот эта команда уничтожит вашу систему. Понятно, что желательно ограничить его права. Поэтому вместо NOPASSWD нужно указать PASSWD и определить список команд, которые вы хотите разрешить пользователю:

```
admin localhost = NOPASSWD: /bin/lprm, PASSWD: /bin/kill, /sbin/shutdown, /sbin/reboot, /
sbin/mount
```

Без пароля можно вводить только команду lprm с правами root — она удаляет задание печати. Остальные команды, заданные после PASSWD, нужно вводить с паролем. Пользователь не имеет права ввести команду, не указанную ни в списке PASSWD, ни в NOPASSWD. К тому же мы ограничили «место действия» — только localhost, то есть ввод команд, требующих прав root, удаленно, например по SSH, запрещен. А это хорошо, ведь в этом случае нужен физический доступ к серверу, нужно пройти через охрану, попасть в поле зрения видеокамер...

Приведенного списка команд вполне хватит для присмотра за сервером. Пользователь сможет завершить зависший процесс, завершить работу системы, перезагрузить компьютер, подмонтировать носитель данных, например флешку. Остальные операции запрещены.

А теперь разберемся, как использовать sudo пользователю из списка sudoers. Когда нужно выполнить команду, требующую прав root, ее нужно указать как аргумент для команды sudo:

```
sudo mount /dev/sdb1 /mnt/flash
```

Команда sudo выполнит ряд проверок:

- Можно ли вообще использовать sudo?
- Можно ли выполнять команду mount?
- Нужно ли запросить пароль перед выполнением этой команды?

Если вам можно использовать sudo и команду sudo mount, скорее всего, sudo запросит пароль (если, конечно, команда mount не находится в списке NOPASSWD, но что бы она там делала?). Нужно ввести пароль пользователя, а не пароль root! Вот в этом и вся прелесть команды sudo — пароль root не нужно никому сообщать. А нужно определить только, кто и какие операции может выполнять.

Команду su в отличие от sudo может выполнить любой пользователь, но ему нужно знать пароль root — без него ничего не получится. Хотя тут все зависит от настройки системы. Например, во FreeBSD использовать su могут только члены группы wheel, остальные пользователи выполнять su не имеют права. А вот в Linux команду su часто может выполнить любой пользователь. Изменить настройки системы, то есть определить, кто может использовать su, можно в файле /etc/pam.d/su.

Сетевые соединения 10

10.1. Рассматриваемые типы сетевых соединений

Довольно часто пользователи подключаются к Интернету по локальной сети: в офисе или в многоэтажке устанавливается шлюз для доступа к Интернету (компьютер или отдельное аппаратное устройство), а к нему по обычной локальной сети подключатся остальные компьютеры. Настройку шлюза для общего доступа к Интернету мы рассмотрим в главе 28, а в этой главе рассмотрим настройку локальной сети.

Вторым наиболее популярным типом сетевого соединения является DSL-соединение, которое обычно используется для соединения с Интернетом. Ранее наиболее популярным способом подключения к Интернету обычных домашних компьютеров было модемное соединение. Но если раньше оно и могло удовлетворить запросы обычного домашнего пользователя, то для предприятия скорости модема явно не хватало. Поэтому на предприятиях использовались или выделенные модемные подключения или цифровые ISDN-соединения. В первом случае покупался хороший и дорогой модем (тогда он стоил около 200 долларов), как правило, производства ZyXEL, который мог работать на выделенной линии. Но скорость тоже оставляла желать лучшего, зато было меньше помех (линия-то выделенная) и, следовательно, меньше обрывов соединения. Во втором случае цифровая ISDN-линия позволяла передавать данные со скоростью 64 Кбит/с. По современным меркам — это очень мало, но тогда, особенно на фоне модема, это был шаг вперед.

Спустя некоторое время были модернизированы АТС (старые аналоговые АТС стали цифровыми), что позволило использовать DSL-подключения. По сравнению с модемным и ISDN-соединениями DSL-соединение обеспечивает высокую скорость передачи данных при низкой стоимости. Раньше минимальная скорость DSL-подключения была 128 Кбит/с, сегодня скоростью в 10 Мбит/с никого не удивишь, при этом в месяц (за безлимитное подключение) нужно платить около 15–20 долларов (для физических лиц, для юридических, как правило, в два раза больше, хотя почему так, я даже не догадываюсь, наверное, из-за налогов).

Но скорость и дешевизна — не единственные преимущества DSL-соединения. К преимуществам можно отнести высокую надежность (это вам не обычный модем) и свободный телефон (вы можете одновременно работать в Интернете и разговаривать по телефону).

В случае с ADSL-соединением можно немного сэкономить на подключении:

- Если купить собственный ADSL-модем, можно не платить абонплату за аренду модема (перед покупкой уточните у провайдера, какой именно модем нужен).
- Если подключить ADSL-модем самостоятельно, без помощи специалистов, можно тоже немного сэкономить. ADSL-модем подключается довольно просто. Телефонный кабель подключается к ADSL-сплиттеру. К ADSL-сплиттеру также нужно подключить все параллельные телефоны, которые имеются в помещении. Затем к сплиттеру подключается модем. К компьютеру ADSL-модем подключается с помощью обычного Ethernet-кабеля. Сплиттер обычно входит в комплект поставки, а вот Ethernet-кабеля может не быть. Чтобы не покупать инструмент для его обжимки, лучше купить сразу уже обжатый кабель (длиной минимум один метр).

В этой книге будет рассматриваться настройка двух типов соединения: соединение по локальной сети и DSL-соединение. Первое соединение для нас актуально, поскольку мы все же занимаемся «переездом» сети предприятия на свободное программное обеспечение, а в офисе без локальной сети — никак нельзя. Второе соединение нас интересует как наиболее перспективное — скорее всего, для подключения к Интернету вы будете использовать именно это соединение. Хотя вполне возможно, что в вашем офисе уже будет один общий шлюз (на все арендуемые офисы) и соединение будет производиться по обычной локальной сети. В этом случае нужно обзавестись еще одним сетевым адаптером (для сервера) и приступить к настройке шлюза.

Поскольку мы договорились, что Linux будем устанавливать на рабочие станции, а FreeBSD — на сервер, то мы рассмотрим только настройку локальной сети в Linux, а настройку DSL-соединения рассматривать не будем. Хотя настройка DSL-соединения в Linux осуществляется довольно просто. Я расскажу, какие конфигураторы нужно использовать, а дальше вы справитесь и без моих комментариев.

10.2. Настройка локальной сети в Linux

Настройка сети в Linux, с одной стороны, очень проста, с другой стороны, есть некоторые нюансы. Простота заключается в следующем:

- По умолчанию сетевой интерфейс настроен на использование DHCP (Dynamic Host Configuration Protocol), следовательно, вам нужно только развернуть DHCP-сервер и больше ничего не нужно делать — сеть на всех рабочих станциях будет настроена автоматически.
- Даже если DHCP-сервера нет и вам придется настраивать сеть вручную, на помощь придут графические конфигураторы, позволяющие быстро и просто

настроить сеть — указать необходимые сетевые параметры. Эти конфигураторы сами внесут необходимые изменения в конфигурационные файлы, перезапустят системные службы (сервисы). Все, что вам нужно, — это указать сетевые параметры и нажать кнопку OK.

- В Linux для сетевого интерфейса (Ethernet) используется имя ethN, где N — номер сетевой карты (нумерация начинается с 0). Обычно на рабочей станции только один сетевой адаптер, поэтому вы точно знаете его имя — eth0. В случае с FreeBSD все сложнее: там имя интерфейса зависит от производителя адаптера, следовательно, на разных системах будет разное имя сетевого интерфейса, что не очень хорошо.

Что же касается нюансов, то они заключаются в конфигурационных файлах. Во FreeBSD нужно отредактировать всего один файл — /etc/rc.conf. Вы знаете его название, осталось только произвести редактирование. В Linux все сложно: в каждом дистрибутиве конфигурационные файлы могут называться по-разному. Например, в Debian нужно редактировать /etc/network/interfaces, в openSUSE — /etc/sysconfig/network/ifcfg-имя. Хорошо, что есть графические конфигураторы, позволяющие быстро настроить сеть и не задумываться, в каких файлах хранятся настройки сети.

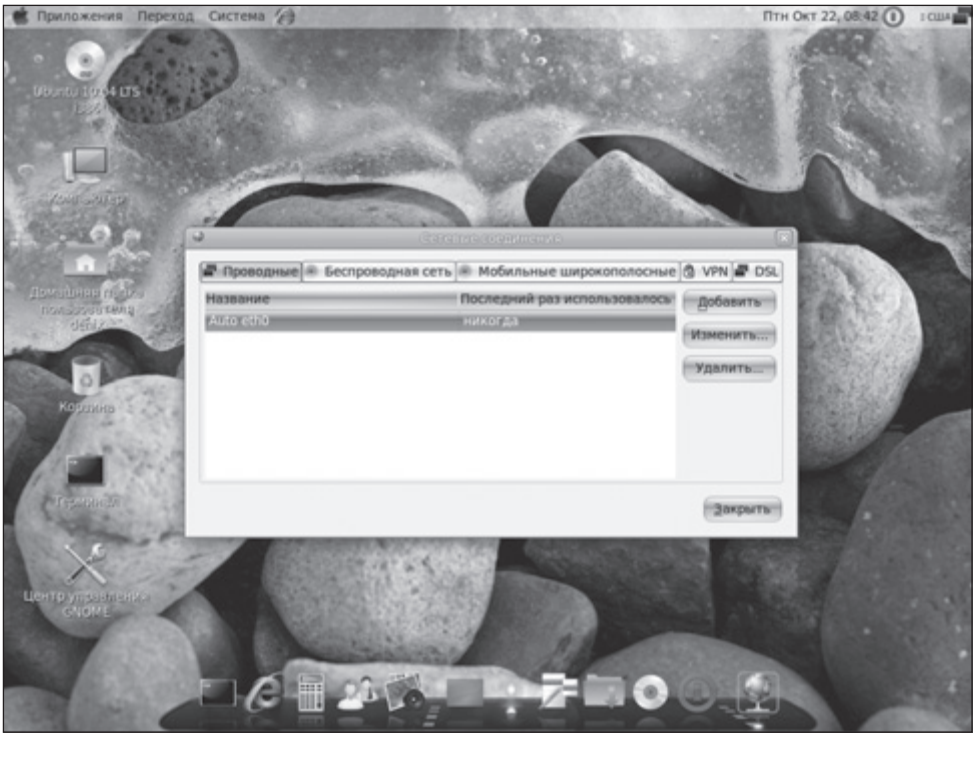

Рис. 10.1. Сетевые соединения

В Fedora, Ubuntu и в некоторых других дистрибутивах (например, Denix, Mint) для настройки сетевых соединений (всех соединений — локальной сети, DSL, VPN) используется Network Manager. Для запуска конфигуратора выполните команду меню Система ▸ Параметры ▸ Сетевые соединения (рис. 10.1). Ethernet-адаптер настраивается на вкладке Проводные. Хотя, повторюсь, при наличии DCHP-сервера настраивать ничего не придется.

В openSUSE сетевое соединение настраивается конфигуратором Сетевые настройки (запускается через конфигуратор YaST — см. рис. 10.2), а DSL-соединение настраивается конфигуратором DSL (тоже запускается через YaST). Для быстрого запуска этих конфигураторов можно использовать команды в терминале:

```
/sbin/yast2 lan
/sbin/yast2 dsl
```

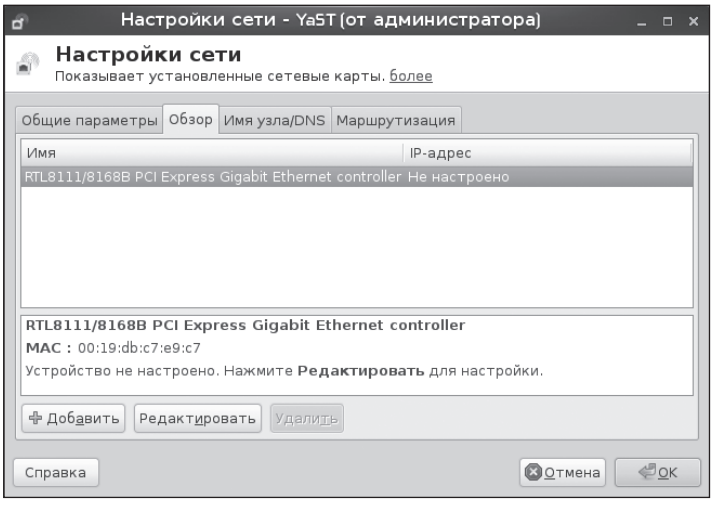

Рис. 10.2. Настройки сети

В Mandriva для настройки сети используется конфигуратор drakconf. Запустите конфигуратор drakconf и перейдите в раздел Сеть и Интернет (рис. 10.3). Не спешите нажимать кнопку Настройка нового сетевого интерфейса. Интерфейс локальной сети (eth0) настраивается при установке системы, поэтому он уже создан. Сейчас нам, возможно, нужно будет изменить его параметры. Запустите Сетевой центр (рис. 10.4).

Нажмите стрелку вниз возле наименования сетевого интерфейса: вы увидите кнопки Monitor, Настройка и Подключиться. Обратите внимание на оранжевый значок возле наименования сетевого интерфейса: сетевой интерфейс не «поднят» (не активен). Когда вы активируете сетевой интерфейс, значок станет зеленым. Оранжевый значок может быть, если сетевой кабель не подключен или же интерфейс настроен на DHCP, но самого DCHP-сервера в сети нет.

Далее все просто: нажимаем кнопку Настройка и указываем параметры сетевого интерфейса (рис. 10.5).

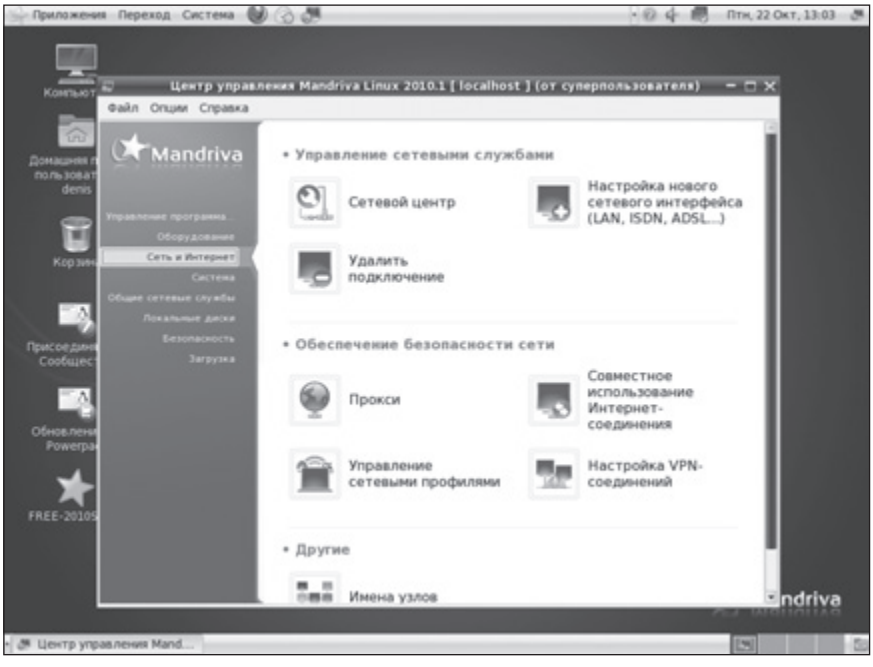

Рис. 10.3. Раздел Сеть и Интернет конфигуратора drakconf

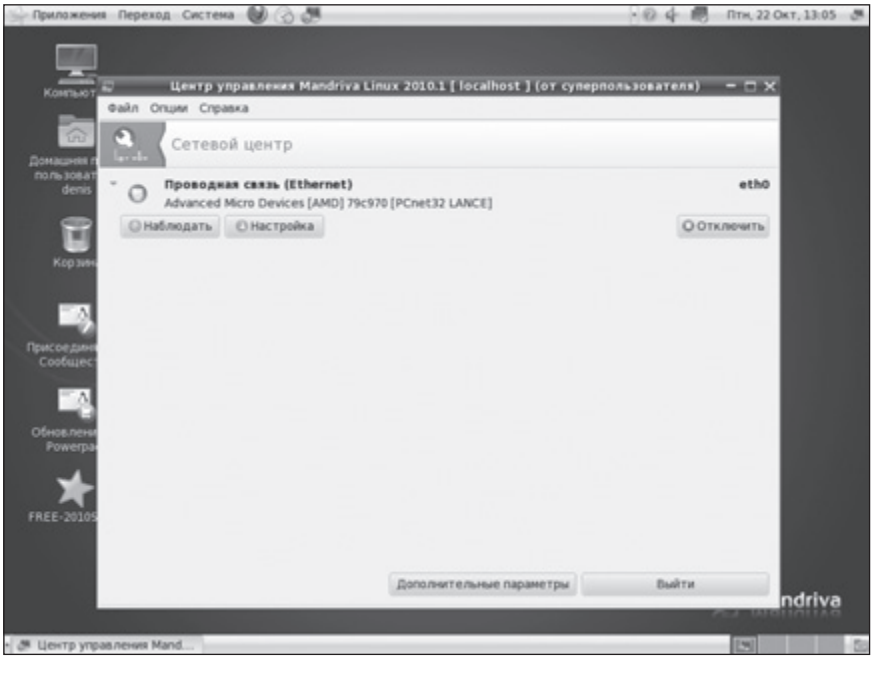

Рис. 10.4. Сетевой центр

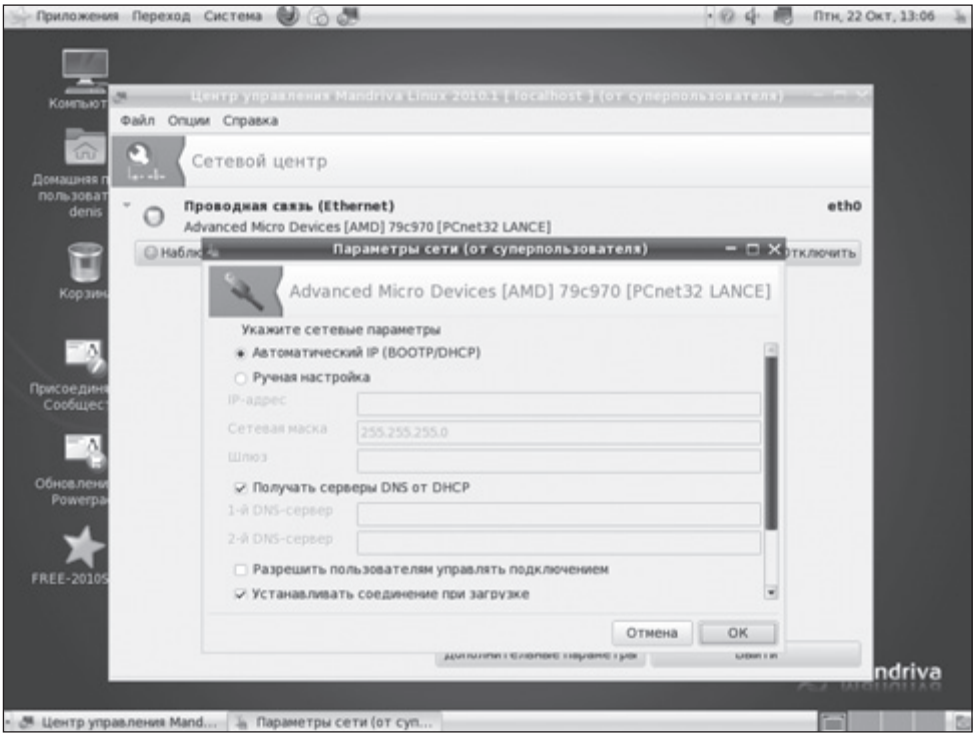

Рис. 10.5. Настройка сетевого интерфейса в Mandriva

10.3. Настройка локальной сети во FreeBSD

10.3.1. Имена сетевых интерфейсов. Автоматическая настройка по DCHP

Как уже было отмечено, во FreeBSD имя сетевого интерфейса зависит от производителя сетевого адаптера. Поэтому в первую очередь вам нужно узнать имя сетевого интерфейса — без этого мы просто не сможем настроить сеть, независимо от того, есть ли в нашей сети DCHP-сервер или нужно настраивать сеть вручную. Определить имя сетевого интерфейса можно командой:

```
# dmesg | grep Ethernet
```

Вывод будет примерно таким:

```
rl0: Ethernet address: XX:XX:XX:XX:XX:XX
```

Здесь rl0 — имя сетевого интерфейса (имя свидетельствует об использовании чипа Realtek 8129 или 8139), а XX:XX:XX:XX:XX:XX — это аппаратный MAC-адрес интерфейса.

У вас может быть совсем другой сетевой адаптер, например адаптер от DEC/Intel, тогда в выводе команды dmesg можно будет обнаружить следующие строки:

```
dc0: <82c169 PNIC 10/100BaseTX> port 0xa000-0xa0ff mem 0xd3800000-0xd38
000ff irq 15 at device 11.0 on pci0
dc0: Ethernet address: 00:a0:XX:XX:XX:XX
...
dc1: <82c169 PNIC 10/100BaseTX> port 0x9800-0x98ff mem 0xd3000000-0xd30
000ff irq 11 at device 12.0 on pci0
dc1: Ethernet address: 00:a0:XX:XX:XX:XX
```

Этот вывод свидетельствует, что в системе имеется два сетевых адаптера (dc0 и dc1), использующих драйвер dc.

Как определить имя интерфейса, мы уже знаем, но ради интереса просмотрите табл. 10.1, в которой приведены соответствия имен интерфейсов во FreeBSD и используемых сетевых адаптеров.

Таблица 10.1. Имена сетевых интерфейсов во FreeBSD

Имя устройства	Сетевая карта
dc	DEC/Intel 21143 и аналогичные адаптеры
vx	3Com 3c590, 3c595
txp	3Com 3cR990
xl	3Com 3c90x
lge	Level 1 LXT1001 Gigabit ethernet (1000 Мбит/с)
rl	RealTek 8129/8139
sis	Silicon Integrated Systems SiS 900/SiS 7016
bfe	Broadcom BCM440x 10/100 Ethernet
pcn	AMD Am79C79x PCI 10/100 NICs
ste	Sundance ST201
sk	SysKonnect SK-984x & SK-982x Gigabit Ethernet (1000 Мбит/с)
sf	Adaptec AIC-6915
bge	Broadcom BCM570xx Gigabit Ethernet (1000 Мбит/с)
de	DEC/Intel DC21x4x
em	Intel PRO/1000 Gigabit Ethernet (1000 Мбит/с)
fxp	Intel EtherExpress Pro/100B (82557, 82558)
ixgb	Intel PRO/10GbE Ethernet
nge	NatSemi DP83820 Gigabit ethernet (1000 Мбит/с)
re	RealTek 8139C+/8169/8169S/8110S
ti	Alteon Networks Tigon I/II Gigabit Ethernet (1000 Мбит/с)
tl	Texas Instruments ThunderLAN
tx	SMC 9432TX
vge	VIA VT612x Gigabit Ethernet (1000 Мбит/с)
vr	VIA Rhine, Rhine II
wb	Winbond W89C840F

Предположим, что наша сеть использует DCHP-сервер. Тогда в файл /etc/rc.conf для настройки сети нужно будет добавить всего одну строку:

```
ifconfig_имя="DHCP"
```

Например:

```
ifconfig_dc0="DHCP"
```

Сохраните файл. Теперь перезапустим сеть, введите команду:

```
# /etc/netstart
```

Давайте посмотрим на вывод сценария netstart:

```
Setting hostuuid: 664d443c-0657-2b41-e2eb-329dfed0124.
Setting hostid: 0xd7721a6c.
Starting Network: lo0 bge0.
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> metric 0 mtu 16384
options=3<RXCSUM,TXCSUM>
inet6 fe80::1%lo0 prefixlen 64 scopeid 0x3
inet6 ::1 prefixlen 128
inet 127.0.0.1 netmask 0xff000000
nd6 options=3<PERFORMNUD,ACCEPT_RTADV>
bge0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
options=9b<RXCSUM,TXCSUM,VLAN_MTU,VLAN_HWTAGGING,VLAN_HWCSUM>
ether XX:XX:XX:XX:XX:XX
inet 192.168.1.121 netmask 0xffffff00 broadcast 192.168.1.255
media: Ethernet autoselect (1000baseT <full-duplex>)
status: active
```

Первым делом нас интересует третья строчка: в ней список настроенных сетевых интерфейсов. В нашем случае их два — интерфейс обратной петли (lo0, loopback), использующийся для тестирования сети, и интерфейс сетевой карты bge0. Имена интерфейсов я каждый раз меняю, чтобы вы привыкали, что они могут быть разными.

Больше всего нас интересует второй интерфейс — bge0. Ему был присвоен IP-адрес 192.168.1.121, текущий режим работы — 1000baseT, полный дуплекс, статус — активен. Все, сеть мы настроили.

10.3.2. Ручная настройка сетевого интерфейса

При автоматической настройке узла DCHP-сервер передает всю необходимую информацию: IP-адрес узла, сетевую маску, имя компьютера, IP-адрес шлюза и IP-адреса серверов DNS. При ручной настройке всю эту информацию вам нужно указать самостоятельно. Первым делом сконфигурируем сетевой интерфейс. Откройте ваш файл /etc/rc.conf. Будем считать, что наш сетевой интерфейс называется bge0. Для его настройки нужно добавить в rc.conf следующие строки:

```
ifconfig_bge0="inet 192.168.1.121 netmask 255.255.255.0"
defaultrouter="192.168.1.1"
hostname="host121.firma.com"
```

Думаю, назначение добавленных строк ясно. В первой строке мы назначаем нашему сетевому интерфейсу адрес 192.168.1.121 (маска сети 255.255.255.0), во второй указываем шлюз по умолчанию, а третья строка задает имя узла. Желательно, чтобы имя узла было зарегистрировано на сервере DNS или «прописано» в файле /etc/hosts. Формат этого файла очень прост:

```
IP-адрес имя имя.домен
```

Например:

```
192.168.1.121 host121 host121.firma.com
```

Осталось только указать IP-адреса серверов DNS. Это можно сделать в файле /etc/resolv.conf:

```
search firma.com
namesever 192.168.1.1
namesever 192.168.1.2
```

В директиве search нужно указать имя своего домена, далее можно указать еще несколько доменов через пробел. Когда вы попытаетесь обратиться к узлу, указав только его имя (без имени домена), то система будет поочередно добавлять к этому имени домены, указанные в директиве search, и пытаться разрешить получившееся доменное имя в IP-адрес.

Как правило, нужно указать два сервера имен — первичный и вторичный. Всего может быть четыре директивы nameserver, в каждой из которых можно указать только один DNS-сервер.

Порядок преобразования (разрешения) доменного имени в IP-адрес такой: сначала система ищет IP-адрес в файле /etc/hosts, а затем уже обращается к DNS-серверу. Учитывая, что файлы /etc/hosts уже давно никто не заполняет, вам, возможно, захочется изменить порядок разрешения имен: чтобы сначала система обращалась к DNS-серверу, а потом пыталась что-то найти в практически пустом файле hosts. Для этого откройте файл /etc/nsswitch.conf и найдите в нем строку, начинающуюся с hosts:

```
hosts: files dns
```

Значения files и dns нужно поменять местами:

```
hosts: dns files
```

В листинге 10.1 приведен измененный файл /etc/nsswitch.conf, необходимые изменения выделены жирным шрифтом.

Листинг 10.1. Модифицированный файл /etc/nsswitch.conf

```
#
# nsswitch.conf(5) - name service switch configuration file
# $FreeBSD: src/etc/nsswitch.conf,v 1.1.10.1 2009/08/03 08:13:06 kensmith Exp $
#
group: compat
group_compat: nis
hosts: dns files
networks: files
passwd: compat
passwd_compat: nis
shells: files
services: compat
services_compat: nis
protocols: files
rpc: files
```

После внесения изменений не забудьте ввести команду /etc/netstart для перезапуска сети.

10.3.3. Программа ifconfig

Программа `ifconfig` используется в UNIX (как во FreeBSD, так и в Linux) для настройки сетевого интерфейса. Если запустить `ifconfig` без параметров, вы увидите список настроенных сетевых интерфейсов:

```
bge0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
options=9b<RXCSUM,TXCSUM,VLAN_MTU,VLAN_HWTAGGING,VLAN_HWCSUM>
ether XX:XX:XX:XX:XX:XX
inet 192.168.1.121 netmask 0xffffff00 broadcast 192.168.1.255
media: Ethernet autoselect (1000baseT <full-duplex>)
status: active
plip0: flags=8810<POINTOPOINT,SIMPLEX,MULTICAST> metric 0 mtu 1500
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> metric 0 mtu 16384
options=3<RXCSUM,TXCSUM>
inet6 fe80::1%lo0 prefixlen 64 scopeid 0x3
inet6 ::1 prefixlen 128
inet 127.0.0.1 netmask 0xff000000
```

Но вывод информации — это далеко не все, что может делать `ifconfig`. Иногда полезно перезапустить сетевой интерфейс. Например, для тестирования DHCP-сервера, чтобы посмотреть, какой адрес будет назначен. Перезапускать всю сеть не хочется (командой `/etc/netstart`), как и перезагружать компьютер. Поэтому введите две команды:

```
# ifconfig имя down
# ifconfig имя up
```

Первая команда закрывает сетевой интерфейс с указанным именем, а вторая — «поднимает» его. Еще такой перезапуск бывает полезным при работе с беспроводными соединениями, когда соединение «зависло», то есть перестало передавать или принимать данные. Сначала нужно попробовать перезапустить интерфейс, а потом — точку доступа.

Также в силах `ifconfig` изменить IP-адрес сетевого интерфейса, например:

```
# ifconfig имя 192.168.1.65 netmask 255.255.255.0
```

Данная команда устанавливает новый IP-адрес узла. При этом изменение IP-адреса осуществляется на лету — не нужно закрывать и поднимать сетевой интерфейс. А вот для изменения MAC-адреса настоятельно рекомендуется сначала закрыть интерфейс, установить новый MAC-адрес, а затем — поднять интерфейс:

```
# ifconfig имя down
# ifconfig имя ether XX:XX:XX:XX:XX:XX
# ifconfig имя up
```

Хотя `ifconfig` и может изменить IP-адрес сетевого интерфейса, рекомендую этим не увлекаться, если в вашей сети есть DHCP-сервер, поскольку назначенный вами IP-адрес уже может быть присвоен другому узлу, что приведет к конфликту IP-адресов.

10.3.4. Таблица маршрутизации

Когда один компьютер отправляет данные другому компьютеру, то все зависит от местоположения второго компьютера. Если он находится в одной сети с первым

компьютером, тогда он просто получит данные. А вот если второй компьютер находится в другой сети, то система должна получить маршрут в другую сеть. Маршруты содержатся в таблице маршрутизации ядра. По сути, маршрут к сети — это адрес маршрутизатора, которому нужно отправить пакеты, а уж он передаст их второму узлу (или другому маршрутизатору — все зависит от структуры сети). Если в таблице маршрутизации нет маршрута к сети, в которой находится получатель данных, тогда данные отправляются по маршруту по умолчанию. А дальше все зависит от шлюза (маршрутизатора) по умолчанию — он может или отправить их второму узлу (если в его таблице маршрутизации будет нужный маршрут) или же отбросить данные, в этом случае отправитель получит сообщение «сеть недоступна».

Мы уже научились устанавливать шлюз по умолчанию. Команда `route` позволяет управлять таблицей маршрутизации на лету — без перезагрузки сети или компьютера. Для установки другого маршрута по умолчанию используются команды:

```
# route delete default
# route add default IP-адрес
```

Сейчас нет смысла подробно говорить о маршрутизации, лучше это сделать в главе 28, когда мы будем рассматривать настройку шлюза.

10.3.5. Конфигуратор sysinstall

В Linux есть конфигураторы сети, во FreeBSD вы можете использовать sysinstall. Правда, когда вы уже знаете, как обойтись без него, вряд ли вы будете его запускать. Но все же я вам расскажу, как настроить сеть с помощью этого конфигуратора. Введите команду:

```
# sysinstall
```

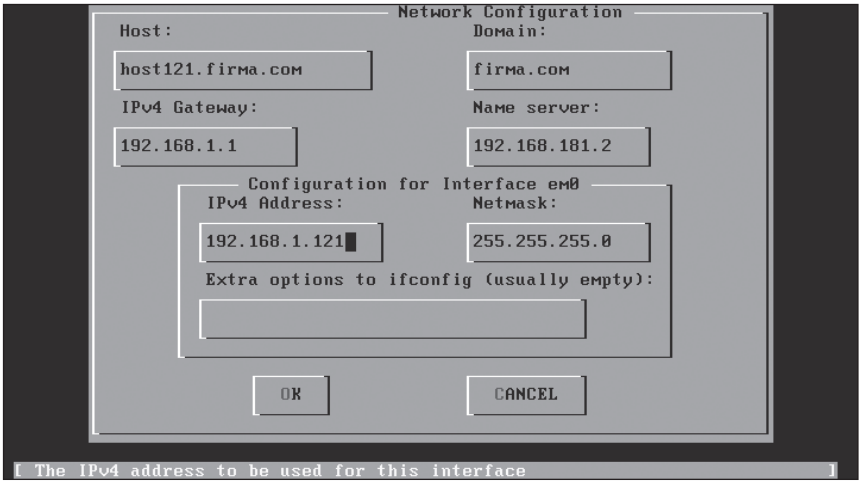

Рис. 10.6. Конфигуратор sysinstall, настройка сети

Далее выберите из меню команду Configure ▸ Networking ▸ Interfaces, после выберите интерфейс, который вы желаете настроить. Конфигуратор уточнит, нужно ли использовать DHCP и IPv6. На оба вопроса стоит, наверное, ответить No, поскольку при использовании DHCP проще добавить одну строчку в /etc/rc.conf, чем бродить по дебрям меню sysinstall, а IPv6 пока не используется широко, поэтому у вас будут обычные IPv4-адреса. Далее нужно указать сетевые параметры (рис. 10.6). Думаю, они в комментариях не нуждаются — и так все понятно.

10.3.6. Диагностика соединения

После настройки соединения (будь то по DCHP или вручную) нужно протестировать его работоспособность. Это делается командой ping. Сначала нужно «пропинговать» только что настроенный интерфейс (в нашем случае это 192.168.1.121):

```
# ping -c5 192.168.1.121
PING 192.168.1.121 (192.168.1.121): 56 data bytes
64 bytes from 192.168.1.121: icmp_seq=0 ttl=64 time=0.082 ms
64 bytes from 192.168.1.121: icmp_seq=1 ttl=64 time=0.074 ms
64 bytes from 192.168.1.121: icmp_seq=2 ttl=64 time=0.076 ms
64 bytes from 192.168.1.121: icmp_seq=3 ttl=64 time=0.108 ms
64 bytes from 192.168.1.121: icmp_seq=4 ttl=64 time=0.076 ms
--- 192.168.1.121 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.074/0.083/0.108/0.013 ms
```

Параметр -c5 задает количество ICMP-пакетов, которые будут отправлены узлу, указанному в качестве второго параметра (192.168.1.121). Пяти пакетов вполне достаточно, чтобы убедиться в работоспособности соединения. Большее количество пакетов нужно для более подробной статистики, если пакеты то доходят, то нет или же есть большие задержки при передаче пакетов.

В этом случае, как говорит статистика, было отправлено 5 пакетов и было получено 5 ответов от «пингуемого узла». То есть вроде бы сетевой интерфейс настроен. Но это еще не все, нам еще нужно «пропинговать» любой другой компьютер локальной сети (лучше всего «пинговать» шлюз сети — он всегда доступен):

```
# ping -c5 192.168.1.1
PING 192.168.1.1 (192.168.1.1): 56 data bytes
64 bytes from 192.168.1.1: icmp_seq=0 ttl=64 time=0.082 ms
64 bytes from 192.168.1.1: icmp_seq=1 ttl=64 time=0.074 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=64 time=0.076 ms
64 bytes from 192.168.1.1: icmp_seq=3 ttl=64 time=0.108 ms
64 bytes from 192.168.1.1: icmp_seq=4 ttl=64 time=0.076 ms
--- 192.168.1.1 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.074/0.083/0.108/0.013 ms
```

Теперь проверим работу сервера DNS и интернет-соединения (при условии, что доступ к Интернету осуществляется по локальной сети). Для этого нужно

«пропинговать» любой узел Интернета, но обратиться к нему нужно по имени, а не по IP-адресу, чтобы убедиться в работоспособности сервера DNS[1]:

```
PING piter.com (92.53.112.56): 56 data bytes
64 bytes from 92.53.112.56: icmp_seq=0 ttl=128 time=70.256 ms
64 bytes from 92.53.112.56: icmp_seq=1 ttl=128 time=72.227 ms
64 bytes from 92.53.112.56: icmp_seq=2 ttl=128 time=71.617 ms
64 bytes from 92.53.112.56: icmp_seq=3 ttl=128 time=70.167 ms
64 bytes from 92.53.112.56: icmp_seq=4 ttl=128 time=70.253 ms
--- piter.com ping statistics ---
5 packets transmitted, 5 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 70.167/70.904/72.227/0.854 ms
```

Здесь возможны варианты. Вы можете получить вывод, подобный представленному ранее, — все нормально, настройка сети завершена. А можете получить сообщение, что невозможно разрешить имя узла piter.com. После этого «пропингуйте» IP-адрес 92.53.112.56[2]: если он доступен, то причина только в DNS. А вот если этот адрес недоступен, то причина и в шлюзе (или в настройках сети) — ведь у вас нет доступа к Интернету, и в DNS-сервере (так как имя разрешить не удалось).

Причин сбоя может быть множество, вот с чего нужно начать:

- Проверьте настройки этого компьютера: указаны ли DNS-серверы в resolv.conf и «прописан» ли маршрут по умолчанию. Если все правильно и указаны правильные IP-адреса DNS-серверов и шлюза, тогда перезапустите сеть сценарием /etc/netstart. В случае отсутствия результата нужно продолжить поиск проблемы далее.
- «Пропингуйте» еще раз шлюз — вы должны убедиться, что он доступен. Если шлюз доступен, то причина только в нем. Возможно, он вообще еще не настроен как шлюз (см. главу 28) или же временно недоступно интернет-соединение (все бывает — нужно звонить провайдеру).

10.3.7. Суперсервер inetd

Далее в этой книге мы будем настраивать сетевые сервисы — веб-сервер (Apache), FTP-сервер и т. д. Сетевой сервис (часто употребляются и другие названия — демон, служба) может запускаться автономно и через суперсервер inetd. У каждого типа запуска есть свои преимущества и недостатки. При автономном запуске программа-сервер запускается при запуске системы. Она начинает пожирать ресурсы системы с момента своего запуска и до завершения работы

[1] Лучше для тестирования использовать не какой-нибудь «общеизвестный» сервер, а тот, у которого знакомый лично вам администратор, с которым вы заранее договоритесь (лучше по обычному телефону), что «сейчас я буду „пинговать" твой сервер». Причина проста — часто большое количество подобных ICMP-запросов (ping) «без предупреждения» может означать подготовку атаки на сервер, и вашему IP-адресу вполне могут, в лучшем случае, вообще заблокировать доступ к этому «общеизвестному» серверу. Вполне возможно — автоматически, без участия администраторов. В худшем — начнут выяснять источник и причину «атаки».

[2] Кстати, нет никаких гарантий, что на момент чтения книги именно вами сохранится этот конкретный ip-адрес для узла piter.com — еще один аргумент в пользу тестирования на серверах со знакомыми администраторами.

системы. Такая схема оправдывает себя, только если программа-сервер часто используется. К примеру, у вас есть корпоративная база данных, доступная через веб-интерфейс, и две сотни пользователей, которые часто обращаются к ней. В этом случае автономно запущенный сервер будет работать быстрее, чем сервер, запускаемый через inetd. Ведь при запуске через inetd запускается только inetd, который «слушает» все порты, указанные в его конфигурационном файле. Когда поступает пакет на определенный порт (в нашем случае это порт 80), он запускает связанную с этим портом программу-сервер и передает ей запрос. Обработав запрос, программа-сервер не остается в памяти, а завершает работу. Если сервер запущен автономно, то он всегда находится в памяти и не нужно при каждом запросе запускать его через inetd, что снижает нагрузку на систему и увеличивает ее производительность.

Но некоторые сервисы целесообразно запускать через inetd. К таким сервисам относится, например, SSH. Нет необходимости запускать его автономно, если им будете пользоваться только вы лично и то не чаще, чем пару раз в месяц, чтобы зайти на сервер удаленно и подправить какой-то конфигурационный файл. Ведь BSD-серверы настраиваются по принципу «настроил и забыл», поэтому вряд ли вы будете часто заходить по SSH на сервер. Поэтому незачем программе sshd работать круглосуточно в ожидании чуда — «пришествия администратора». А по сему лучше настроить ее запуск через inetd. Прошу заметить, что сейчас мы не будем говорить о настройке SSH, мы просто рассмотрим настройку запуска через inetd на примере SSH.

Главный конфигурационный файл inted называется /etc/inet.conf. Откройте его (листинг 10.2).

Листинг 10.2. Конфигурационный файл /etc/inet.conf

```
# Сервис  Сокет   Протокол  W/NW    Пользователь  Программа              Аргументы
ftp       stream  tcp       nowait  root          /usr/libexec/ftpd      ftpd -l
ftp       stream  tcp6      nowait  root          /usr/libexec/ftpd      ftpd -l
#ssh      stream  tcp       nowait  root          /usr/sbin/sshd         sshd -i -4
#ssh      stream  tcp6      nowait  root          /usr/sbin/sshd         sshd -i -6
#telnet   stream  tcp       nowait  root          /usr/libexec/telnetd   telnetd
#telnet   stream  tcp6      nowait  root          /usr/libexec/telnetd   telnetd
#shell    stream  tcp       nowait  root          /usr/libexec/rshd      rshd
#shell    stream  tcp6      nowait  root          /usr/libexec/rshd      rshd
#login    stream  tcp       nowait  root          /usr/libexec/rlogind   rlogind
#login    stream  tcp6      nowait  root          /usr/libexec/rlogind   rlogind
#finger   stream  tcp       nowait  nobody        /usr/libexec/fingerd   fingerd -s
#finger   stream  tcp6      nowait  nobody        /usr/libexec/fingerd   fingerd -s
#
# используется для получения уведомлений о новой почте на консоль
#comsat   dgram   udp       wait    tty:tty       /usr/libexec/comsat    comsat
#
# ntalk нужен для работы утилиты 'talk' (см. man talk)
#ntalk    dgram   udp       wait    tty:tty       /usr/libexec/ntalkd    ntalkd
#tftp     dgram   udp       wait    root          /usr/libexec/tftpd     tftpd -l -s /tftpboot
#tftp     dgram   udp6      wait    root          /usr/libexec/tftpd     tftpd -l -s /tftpboot
#bootps   dgram   udp       wait    root          /usr/libexec/bootpd    bootpd
#
# Стандартный набор "небольших" серверов
```

продолжение ⇩

Листинг 10.2 *(продолжение)*

```
# Если вам нужен какой-то сервер, расскомментируйте соответствующую строчку
#
#daytime  stream  tcp       nowait  root          internal
#daytime  stream  tcp6      nowait  root          internal
#daytime  dgram   udp       wait    root          internal
#daytime  dgram   udp6      wait    root          internal
#time     stream  tcp       nowait  root          internal
#time     stream  tcp6      nowait  root          internal
#time     dgram   udp       wait    root          internal
#time     dgram   udp6      wait    root          internal
#echo     stream  tcp       nowait  root          internal
#echo     stream  tcp6      nowait  root          internal
#echo     dgram   udp       wait    root          internal
#echo     dgram   udp6      wait    root          internal
#discard  stream  tcp       nowait  root          internal
#discard  stream  tcp6      nowait  root          internal
#discard  dgram   udp       wait    root          internal
#discard  dgram   udp6      wait    root          internal
#chargen  stream  cp        nowait  root          internal
#chargen  stream  tcp6      nowait  root          internal
#chargen  dgram   udp       wait    root          internal
#chargen  dgram   udp6      wait    root          internal
#
# CVS-серверы - для главного CVS-репозитория
#
#cvspserver  stream  tcp   nowait   root    /usr/bin/cvs    cvs --allow-root=/your/cvsroot/here pserver
#cvspserver  stream  tcp   nowait   root    /usr/bin/cvs    cvs --allow-root=/your/cvsroot/here kserver
#
# RPC-сервисы, для их работы нужен запущенный rpcbind
#
#rstatd/1-3    dgram  rpc/udp  wait  root    /usr/libexec/rpc.rstatd         rpc.rstatd
#rusersd/1-2   dgram  rpc/udp  wait  root    /usr/libexec/rpc.rusersd        rpc.rusersd
#walld/1       dgram  rpc/udp  wait  root    /usr/libexec/rpc.rwalld         rpc.rwalld
#pcnfsd/1-2    dgram  rpc/udp  wait  root    /usr/local/libexec/rpc.pcnfsd   rpc.pcnfsd
#rquotad/1     dgram  rpc/udp  wait  root    /usr/libexec/rpc.rquotad        rpc.rquotad
#sprayd/1      dgram  rpc/udp  wait  root    /usr/libexec/rpc.sprayd         rpc.sprayd
#
# POP3-сервер
#
#pop3     stream  tcp    nowait    root    /usr/local/libexec/popper    popper
#
# IMAP-сервер
#
#imap4    stream  tcp    nowait    root    /usr/local/libexec/imapd    imapd
#
# NNTP-сервер
#
#nntp     stream  tcp    nowait    news    /usr/local/libexec/nntpd    nntpd
#
# UUCP-сервер
#
#uucpd    stream  tcp    nowait    root    /usr/local/libexec/uucpd uucpd
#
#auth     stream  tcp    nowait    root    internal
```

```
#auth     stream  tcp6   nowait    root    internal
#
# Настоящий "ident"-сервис с поддержкой ~/.fakeid.
#
#auth     stream  tcp    nowait    root    internal    auth -r -f -n -o UNKNOWN -t 30
#auth     stream  tcp6   nowait    root    internal    auth -r -f -n -o UNKNOWN -t 30
#
# Внешний ident-сервер
#
#auth     stream  tcp    wait      root    /usr/local/sbin/identd    identd -w -t120
#
# Запуск qmail (альтернативный MTA) через inetd
#
#smtp     stream  tcp    nowait    qmaild  /var/qmail/bin/tcp-env    tcp-env /var/qmail/bin/qmail-smtpd
#
# Поддержка Samba/SWAT (веб-интерфейс для настройки Samba) через inetd
#
#netbios-ssn  stream    tcp    nowait        root    /usr/local/sbin/smbd    smbd
#netbios-ns   dgram     udp    wait          root    /usr/local/sbin/nmbd    nmbd
#swat         stream    tcp    nowait/400    root    /usr/local/sbin/swat    swat
```

Приведенный в листинге 10.1 конфигурационный файл я немного изменил: удалил лишние комментарии, добавил описание формата, а оставшиеся комментарии перевел на русский язык. Жирным выделены строки, относящиеся к SSH (понятно, что их нужно раскомментировать, если вам нужен SSH-сервер):

```
#ssh    stream    tcp    nowait    root    /usr/sbin/sshd    sshd -i -4
#ssh    stream    tcp6   nowait    root    /usr/sbin/sshd    sshd -i -6
```

Формат конфигурационного файла задан в самом первом комментарии:

```
# Сервис  Сокет  Протокол W/NW     Пользователь   Программа     Аргументы
```

Сначала указывается имя сервиса (в нашем случае — это ssh), затем — тип сокета. Для протокола TCP тип сокета всегда будет stream (поток), а вот для протокола UDP — dgram (дейтаграмма). Далее указывается протокол. Первая запись обеспечивает поддержку самого обычного SSH, а вот вторая — SSH с поддержкой IPv6. Следующее поле может содержать либо wait, либо nowait. Для протокола TCP обычно указывается значение nowait, а для протокола UDP — wait. После этого поля следует имя пользователя, от имени которого будет запущена программа-сервер (обычно это root). Следующие два поля — это полный путь к программе-серверу и передаваемые ей параметры.

Все как бы понятно, но в inetd.conf не указывается номер порта для нашего сервера. А это потому, что номер порта указан в файле /etc/services. Я не буду приводить этот файл полностью, а приведу только фрагмент, чтобы продемонстрировать его формат (листинг 10.3).

Листинг 10.3. Фрагмент файла /etc/services

```
ftp     21/tcp          #File Transfer [Control]
ftp     21/udp          #File Transfer [Control]
ssh     22/tcp          #Secure Shell Login
ssh     22/udp          #Secure Shell Login
telnet  23/tcp
telnet  23/udp
```

В первом поле указывается имя сервиса, оно должно быть таким же, как в файле /etc/inetd.conf, далее указывается номер порта и протокол.

10.4. Настройка DSL-соединения во FreeBSD

Настройка DSL-соединения в FreeBSD осуществляется так же просто, как и сетевой карты, конечно, при условии отсутствия проблем с оборудованием. В большинстве случаев вам нужно отредактировать всего один конфигурационный файл — /etc/ppp/ppp.conf (листинг 10.4).

Листинг 10.4. Файл /etc/ppp/ppp.conf

```
default:
    set device PPPoE:bge0
    set speed sync
    set mru 1492
    set mtu 1492
    set ctsrts off
    enable dns
internet:
    set authname имя
    set authkey пароль
    add default HISADDR
```

Значения, выделенные жирным, вам нужно заменить собственными:

- bge0 — имя сетевого интерфейса, у вас будет другим;
- имя — имя пользователя (можно посмотреть в договоре с провайдером на предоставление услуг);
- пароль — пароль пользователя;
- HISADDR — в большинстве случаев изменять это значение не нужно, поскольку HISADDR означает, что IP-адрес шлюза по умолчанию будет назначен сервером провайдера при установке соединения.

Параметры mru и mtu устанавливают соответственно максимальный размер принимаемого и передаваемого пакета. Значение 1492 рекомендуется как оптимальное для этого типа соединения. Строка «enable dns» запрашивает у сервера провайдера адреса DNS-серверов.

По большому счету, представленный в листинге 10.4 конфигурационный файл готов к использованию. Но, возможно, вам захочется добавить в секцию default параметры управления переподключением соединения, что особенно актуально для сервера (дабы в случае сбоя соединение оно было установлено автоматически):

```
enable lqr echo
set lqrperiod 60
set reconnect 5 10
set redial 10 3
```

Первые две строки включают запрос проверки качества (Link Quality Request) и устанавливают период проверки 60 секунд. Каждые 60 секунд система будет

проверять качество соединения. Если произошел разрыв соединения, оно будет установлено автоматически.

Параметр reconnect обеспечивает автоматическое переподключение соединения, максимальное количество попыток — 10, пауза между попытками — 5 секунд. Параметр redial управляет «дозвоном» к провайдеру, то есть вступает в силу при начальной установке соединения (а не при переподключении). Возможно, сервер провайдера временно недоступен или возникла другая проблема. В этом случае будет сделано всего три попытки «дозвониться», пауза между которыми составит 10 секунд.

Осталось только запустить соединение:

```
# ppp -ddial internet
```

После этого попробуйте «пропинговать» удаленный узел, например,

```
ping mai.ru
```

Если «пинг» неудачен, соединение не было установлено, причину нужно искать в журналах /var/log/messages и /var/log/ppp.log.

Осталось только настроить систему для автоматического подключения к Интернету при запуске системы. Для этого в rc.conf добавьте строки:

```
ppp_enable="YES"
ppp_mode="ddial"
ppp_profile="internet"# измените имя профиля (см. ppp.conf)
```

Установка программного обеспечения в Linux

11

11.1. Понятие о пакете

11.1.1. Отличие пакета от инсталлятора Windows-программ

В Windows мы привыкли устанавливать программы так: запускаем установочный файл (как правило, он называется `setup.exe` или `install.exe`), затем отвечаем на вопросы мастера установки (в какой каталог установить программу, как будет называться программная группа, создавать ли ярлыки на рабочем столе и т. д.). Дистрибутив Windows-программы — обычно это набор из нескольких файлов: установочный файл и архив (или архивы), содержащий все файлы программы. Иногда это один установочный файл, и все необходимое для установки программы находится в нем (по сути, в этом случае установочный файл является самораспаковывающимся архивом с функциями мастера установки).

В Linux все иначе. Все файлы, относящиеся к программе, которую мы устанавливаем, и сама программа находятся в пакете. Отвечает за установку пакета специальная программа — менеджер пакетов. С помощью менеджера пакетов можно установить, удалить и обновить пакет.

Пакет содержит не только файлы программы, но и инструкции для менеджера пакетов — куда установить программу, куда записать конфигурационные и вспомогательные файлы. Возможно, придется выполнить дополнительные действия, например создать учетную запись пользователя, от имени которого будет работать программа (это характерно для сетевых серверов Linux). Вы не можете вмешаться в процесс установки программы. Все, что вы можете сделать, — это задать операцию над пакетом, например установка. Все остальное за вас сделает менеджер пакетов. С одной стороны, удобно — зачем отвечать на глупые вопросы мастера установки программы в Windows, если в большинстве случаев мы просто нажимаем кнопку Далее, оставляя все параметры установки как есть. С другой стороны, не очень удобно. Все пакеты устанавливаются в корневую файловую систему.

Если у вас на ней мало места, программу вы не установите. А ведь в Windows можно было просто выбрать другой раздел (диск). Но в Linux это так, и ничего вы не сделаете. Максимум, что вы можете, — просто знать об этом.

Прежде чем приступить к практике мы должны поговорить о формате пакетов и зависимостях пакетов.

11.1.2. Форматы пакетов в Linux

Дистрибутивы, так или иначе относящиеся к Red Hat (Fedora, CentOS, Mandriva, openSUSE и т. д.), используют пакеты формата RPM. Много разных RPM-пакетов доступно по адресу http://rpm.pbone.net и http://rpmfind.net. Кроме формата RPM есть еще формат DEB, который используется в Debian-совместимых дистрибутивах (Debian, Ubuntu, Mint, Denix и т. д.)

Оба формата (RPM и DEB) хороши, а Linux-программы часто доступны как в DEB-пакетах, так и в RPM-пакетах. Просто вы должны знать, что если где-то придется скачать пакет с программой, то для Fedora/Mandriva/openSUSE нужно скачивать RPM-пакет, а не DEB. Узнать где какой пакет просто — у RPM-пакетов «расширение» .rpm, а у DEB-пакетов — .deb. Слово «расширение» в кавычках потому, что в Linux нет такого понятия, как «расширение файла», но слово «расширение» воспринимается проще, чем «последние три символа имени файла».

11.1.3. Зависимости и конфликты

Некоторые пакеты для своей работы требуют наличия других пакетов. Например, пакет с игрой может требовать пакета, содержащего графическую библиотеку. Это понятно: без этой библиотеки игра не запустится. Существует теоретическая возможность установки пакетов без удовлетворения зависимостей, но нет никакой гарантии, что установленные программы будут работать. Раньше разрешать зависимости нужно было вручную. Менеджер пакетов просто выдавал список файлов, которые нужны для установки пакета, а вы уже сами определяли, какие пакеты содержат данные файлы, и устанавливали пакеты вручную. Понятно, что это очень неудобно, а установка программы из-за этого занимала много времени, даже если сама программа очень небольшого размера. Современные менеджеры пакетов сами разрешают зависимости пакетов и устанавливают все необходимое. Конечно, они предупреждают вас о том, что сейчас недостающие пакеты будут загружены из Интернета (если таких пакетов нет на диске) и что они занимают столько-то места. Вдруг вы передумаете устанавливать программу, если узнаете, что из Интернета нужно скачать дополнительные пакеты общим объемом 100 Мбайт.

Существуют и «обратные зависимости», то есть конфликты. Иногда устанавливаемый пакет конфликтует с уже установленным пакетом. Вам тогда нужно решить, удалить ли уже установленный пакет или же не устанавливать выбранный вами пакет. Как правило, конфликтуют пакеты программ, выполняющих одни и те же или взаимно обратные действия. Сейчас поясню. Далеко не всегда будут конфликтовать пакеты программ, которые выполняют одни и те же действия. Например, в вашей системе может быть установлено огромное количество текстовых редакторов. Все они выполняют одну и ту же функцию, но не мешают

друг другу при этом. А иногда программы, которые выполняют одну и ту же функцию, могут мешать друг другу работать. Вот это и есть конфликт пакетов. Например, в системе может быть только один почтовый агент (не путайте с почтовым клиентом), то есть программа, организующая передачу почты на другой сервер. Поэтому если вам нужен MTA (Mail Transfer Agent), вам придется выбирать между postfix и sendmail — это самые популярные MTA в Linux. Вообще, данная ситуация скорее исключение из правил, чем правило, да и MTA не нужен домашнему пользователю — это удел сервера.

11.1.4. Репозитории — хранилища пакетов

Хранилища пакетов называются репозиториями. Репозиторий может быть локальным, например находиться на локальном жестком диске или на DVD-диске, а может быть сетевым. В последнем случае пакеты находятся на сервере в Интернете[1].

Репозиторий решает проблему централизованной установки и обновления пакетов. Например, менеджер пакетов может проверить сетевые репозитории и сообщить вам, какие пакеты пора обновить. Да и установка пакетов из репозитория выполняется значительно проще, потому что у менеджера пакетов есть список доступных пакетов, и он, зная, где находится тот или иной пакет, может самостоятельно разрешать зависимости пакетов.

11.1.5. Доступные менеджеры пакетов

Так уж получилось, что хотя разные дистрибутивы и используют один и тот же формат пакетов, менеджеры пакетов (программы, управляющие пакетами) у них разные. В табл. 11.1 представлены самые актуальные менеджеры пакетов.

Таблица 11.1. Менеджеры пакетов

Название программы	Описание
rpm	Самая древняя программа. Используется во всех дистрибутивах, которые поддерживают формат RPM (Red Hat, Fedora, CentOS, openSUSE). Работает в текстовом режиме и не умеет разрешать зависимости, то есть доустанавливать необходимые пакеты. Чтобы установить пакет A, который требует наличия пакетов B и C, вам нужно сначала вручную установить пакеты B и C, а уж после этого устанавливать пакет A
urpmi	Работает в текстовом режиме, но поддерживает репозитории, следовательно, умеет разрешать зависимости. Используется в дистрибутиве Mandriva
rpmdrake	Графический менеджер пакетов, умеющий разрешать зависимости и настраивать репозитории. Все операции, связанные с программным обеспечением (от добавления репозитория до установки пакетов), можно выполнить с помощью этой программы. Используется в Linux Mandriva
yum	Работает в текстовом режиме, поддерживает репозитории и умеет разрешать зависимости. Фактически то же, что и urpmi, но используется в Fedora, ASP Linux и некоторых других дистрибутивах

[1] Или в локальной сети — в основном различия могут быть лишь в способе адресации и скорости доступа.

Название программы	Описание
gpk-application	Графический менеджер пакетов, используемый в Fedora. В старых версиях использовался менеджер пакетов pirut, он же system-config-packages
dpkg	Аналог rpm, но в Debian-совместимых дистрибутивах. Простейший менеджер пакетов, не поддерживающий репозитории и не умеющий разрешать зависимости
apt-get	Используется преимущественно в Debian-совместимых дистрибутивах, умеет разрешать зависимости и поддерживает репозитории. Работает в текстовом режиме
aptitude	Обладает псевдографическим интерфейсом пользователя: работает в текстовом режиме, но имеет интерфейс пользователя с окошками и меню. В остальном похожа на apt-get
synaptic	Графический менеджер пакетов, используется с Ubuntu и дистрибутивах, основанных на нем, — Mint, Denix
zypper	Используется в openSUSE, работает в текстовом режиме, поддерживает репозитории и умеет разрешать зависимости

Графические менеджеры пакетов мы вообще не будем рассматривать. В этом просто нет смысла. Главное, что вы знаете, как они называются, а дальше разберетесь сами. Неужели так трудно выбрать пакеты и нажать кнопку Установить? Именно по такому принципу и работают все графические менеджеры. А вот с программами, работающими в текстовом режиме, все не так просто. Тут важно знать описание параметров этих программ, чтобы выполнить необходимые вам действия. Если вы будете устанавливать Linux на сервере, то вам особенно важно знать, как работают обычные (не графические) программы; на сервере частенько не устанавливают графическую подсистему — в ней нет необходимости.

Из программ, которые работают в текстовом режиме, мы рассмотрим rpm, zypper, dpkg и apt-get. Программа RPM заслуживает внимания, поскольку она используется во всех RPM-совместимых дистрибутивах. То же самое можно сказать о dpkg — она используется в Debian, Ubuntu, Mint, Denix и других дистрибутивах. Даже если вы не знаете, какой основной менеджер пакетов используется в том дистрибутиве, вы сможете использовать программы rpm или dpkg (в зависимости от используемого формата пакетов). Программа zypper используется в openSUSE (просто мне нравится этот дистрибутив), а программа apt-get используется во всех Debian-совместимых дистрибутивах и не только. Именно поэтому я выбрал эти четыре программы. Почему не рассматриваются yum и urpmi? Потому что в дистрибутивах Fedora и Mandriva вы будете чаще использовать графические конфигураторы — gpk-application и rpmdrake соответственно.

11.2. Программа rpm

Использовать программу rpm довольно просто. Только помните о том, что она не поддерживает разрешение зависимостей, поэтому если пакет A требует наличия пакета B, вы увидите соответствующее сообщение и не сможете установить пакет A до тех пор, пока не установите пакет B.

Для установки пакета нужно ввести команду:

```
# rpm -ihv <пакет.rpm>
```

Для удаления пакета используется опция -e, причем указывать «расширение» не нужно:

```
# rpm -e <пакет>
```

Просмотреть список всех установленных пакетов можно с помощью команды:

```
# rpm -qa | less
```

Если вам нужно просмотреть список пакетов, содержащих в названии строку game, введите команду:

```
# rpm -qa | grep game
```

Для обновления пакета используется параметр -U:

```
rpm -U <пакет>
```

11.3. Программа dpkg

Программа dpkg изначально использовалась в дистрибутиве Debian. Позже она «перекочевала» в другие дистрибутивы, которые основаны на Debian, — нужно же обеспечить полную совместимость с Debian. Программа dpkg довольно стара, поэтому не умеет разрешать зависимости и не поддерживает источники пакетов.

Предположим, что у нас уже есть .deb-пакет. Возможно, вы его скачали в Интернете самостоятельно или нашли на каком-то компакт-диске. Одним словом, вы хотите его установить. Причем хотите установить именно этот пакет, который есть у вас на диске, а не тот, который менеджер пакетов загрузит из Интернета.

Для этого используется программа dpkg. Точнее она используется не только для установки, но и для удаления пакетов и управления ними. Данная команда работает в режиме командной строки, поэтому запускать ее придется из терминала. Учитывая, что для ее запуска нужны права root, то синтаксис ее запуска будет выглядеть так:

```
sudo dpkg [ключи] пакеты
```

Для установки пакета используется ключ -i, например, для установки пакета нужно выполнить команду:

```
sudo dpkg -i /путь/пакет.deb
```

Поскольку программа не знает, где искать пакет (ведь мы же устанавливаем его не из репозитория), то нужно указать полный путь к пакету.

Для удаления пакета используется ключ -r. В этом случае нужно указать просто имя пакета (без .deb):

```
sudo dpkg -r пакет
```

Кроме ключей -i и -r вы можете использовать следующие ключи:

- --unpack < пакета.deb> — распаковывает, но не устанавливает пакет, полезно, если нужно извлечь из пакета несколько файлов;
- -L <пакет> — выводит список файлов уже установленного пакета;

- -p <пакет> — выводит информацию о пакете;
- -s <пакет> — выводит статус пакета.

Подробную информацию о данной программе всегда можно получить в справочной системе:

```
man dpkg
```

Как уже было отмечено, программа dpkg не разрешает зависимости пакетов. Например, если устанавливаемый вами пакет требует наличия других пакетов, то нужно вручную их установить до установки нужного вам пакета. Не очень удобно. Поэтому, возможно, вам будет удобнее использовать программу apt-get; например, для установки пакета package нужно выполнить следующую команду:

```
sudo apt-get install package
```

Но данная программа ничем не отличается от графического менеджера (кроме интерфейса, разумеется): она просматривает файл sources.list и загружает нужный вам пакет из Интернета. Чуть позже программа apt-get будет рассмотрена более подробно.

11.4. Программа zypper

Как уже было отмечено, программа zypper работает в текстовом режиме, умеет разрешать зависимости и поддерживает репозитории. Прежде чем рассмотреть опции программы, разберемся, как она работает. Предположим, что вам нужно установить пакет А. Программа zypper обращается к репозиториям (источникам пакетов), список которых хранится в каталоге /etc/zypp/repos.d, и получает списки пакетов. В одном из списков пакетов будет найден наш пакет. Программа получает его (просто скачивает с удаленного сервера пакетов), но не устанавливает. Если для работы пакета нужны дополнительные пакеты Б и В, то аналогично zypper находит эти пакеты в списках, загружает и устанавливает их, а уж потом устанавливает наш пакет А. Вся работа по установке пакета автоматизирована — вам нужно только подтвердить установку дополнительных пакетов. Лишь бы только было соединение с Интернетом — ведь без этого нельзя получить списки пакетов и установить их. Хотя, zypper поддерживает и локальные репозитории в виде DVD с пакетами. Все зависит от настроек в каталоге /etc/zypp/repos.d — каждому источнику пакетов соответствует отдельный файл в этом каталоге. Вот пример описания источника пакетов:

```
[openSUSE-11.3-DVD 11.3]
name=openSUSE-11.3-DVD 11.3
enabled=1
autorefresh=0
baseurl=cd:///?devices=/dev/sr0
path=/
type=yast2
gpgcheck=1
```

Параметр name — это просто имя источника пакетов. Параметр enabled установлен в 1, значит, источник пакетов активен. Параметр autorefresh (автоматическое обновление) имеет смысл только для сетевых репозиториев, которые могут со временем обновляться (могут быть добавлены новые пакеты или обновиться уже имеющиеся), для источника пакетов в виде DVD этот параметр неактуален (поэтому установлен в 0), поскольку файлы на DVD не могут быть обновлены.

Третий параметр, baseurl, задает путь к самим пакетам. В нашем случае такой путь подразумевает использование DVD-диска. Параметр path актуален тоже только для DVD, для сетевого источника пакетов он не задается.

Последние два параметра — это тип источника пакетов и GPG-проверка. Для локального источника (DVD-диска) проверку можно выключить: смело можете установить параметр gpgcheck в 0.

Для сетевого источника пакетов еще может быть задан параметр keeppackages. Как уже было отмечено, zypper сначала загружает пакеты из источника пакетов, а потом устанавливает их. Если параметр keeppackages равен 1, то загруженные файлы пакетов не будут удалены после установки пакетов (что удобно, если вы хотите установить эти же пакеты на другом компьютере, но хранение пакетов требует дополнительного дискового пространства). А если параметр keeppackages равен 0, то после установки пакетов исходные файлы пакетов будут удалены для экономии места.

Основной файл конфигурации zypper называется /etc/zypp/zypp.conf, но вы вряд ли будете его редактировать, поскольку все параметры в нем устраивают большинство пользователей. Теперь поговорим об использовании zypper. Синтаксис вызова программы такой:

```
zypper <параметры> [пакеты]
```

Параметры программы zypper описаны в табл. 11.2.

Таблица 11.2. Параметры программы zypper

Параметр	Описание
sl	Выводит список источников пакетов
sa URL имя	Добавляет источник пакетов, вам нужно указать адрес источника пакетов и имя, под которым он будет отображаться в списке: zypper sa адрес имя
sd URL|имя	Удаляет источник пакетов. Нужно указать или адрес (URL) источника пакетов или его имя
install пакеты	Устанавливает пакеты, список пакетов разделяется пробелами. Например: zypper install lynx mutt
search маска	Производит поиск пакета, вы можете использовать символ * для замены произвольного количества символов в имени пакета, если вы не знаете точного имени пакета: zypper search mutt*
list-updates	Отображает список пакетов, доступных для обновления
update пакет	Производит обновление пакета. Если вы забыли указать имя пакета, то произойдет обновление всех пакетов
info пакет	Отображает информацию о пакете
remove пакет	Удаляет пакет

11.5. Программа apt-get

Программа apt-get, как и программа zypper, работает в текстовом режиме, поддерживает разрешение зависимостей и репозитории. Работа с репозиториями осуществляется так же, как и в случае с zypper, только список репозиториев хранится не в отдельных файлах, а в одном большом файле — /etc/apt/sources.list. В других главах этой книги нам придется отредактировать этот файл, чтобы добавить в него дополнительные источники пакетов (мы будем добавлять репозиторий Medibuntu для установки кодеков в Ubuntu).

Загруженные из источника пакетов файлы помещаются в каталог /var/cache/apt/archives, но не удаляются после установки пакетов, поэтому для экономии места на диске время от времени не забывайте очищать этот каталог.

Формат вызова программы apt-get следующий:

```
apt-get [параметры] команды [пакет]
```

Основные команды apt-get приведены в табл. 11.3.

Таблица 11.3. Основные команды программы apt-get

Команда	Описание
update	Обновляет внутреннюю базу данных о пакетах. По сути, выполняется синхронизация этой самой внутренней базы данных с источниками пакетов из файла sources.list
upgrade пакет	Производит обновление указанного пакета. Если пакет не указан, производит обновление всей системы — всех пакетов. Однако для обновления всей системы рекомендуется использовать команду dist-upgrade, а эту команду нужно использовать только для обновления указанного пакета
dist-upgrade	Обновление всей системы
install	Установка пакетов, пакеты указываются через пробел
remove	Удаление пакетов, пакеты также указываются через пробел
check	Запускает проверку базы данных, производится поиск нарушений зависимостей пакетов
clean	Очищает каталог /var/cache/apt/archives

Установка программного обеспечения во FreeBSD

12

12.1. Особенности установки программ во FreeBSD

В предыдущей главе вы познакомились с процессом установки программного обеспечения в Linux. Во FreeBSD программное обеспечение можно также установить из пакетов (только формат пакетов отличается от того, что используется в Linux), из коллекции портов и из исходного кода.

Начнем с последнего варианта. При установке программы из исходного кода вам нужно самостоятельно скачать архив, содержащий исходный код программы, распаковать его в какой-то каталог файловой системы (обычно распаковывают в каталог /usr/src), перейти в этот каталог и выполнить команду `make config install clean`. С одной стороны, ничего сложного. Нужно только самостоятельно найти архив с исходным кодом. С другой стороны, нет никакой гарантии, что исходный код вообще откомпилируется. Ведь разработчик программы не знает особенностей вашей системы — возможно, у вас не хватает каких-то заголовочных файлов или же есть банальная ошибка в самом исходном коде.

Переходим ко второму варианту — установка из коллекции портов. Коллекция портов обычно устанавливается при установке системы в каталог /usr/ports. По сути, это коллекция исходного кода программы. Вы переходите в каталог, содержащий исходный код программы, и вводите ту же команду `make config install clean`. Но в отличие от предыдущего варианта, система самостоятельно загружает исходный код (если этого еще не было сделано) из Интернета. Да и вероятность того, что программа откомпилируется, практически 100%. Для платформы i386 эта вероятность составляет 100%, а вот для AMD64 вероятность успешной сборки порта меньше, поскольку не все порты нормально «собираются» на этой платформе. Таковы реалии, но в будущем все ошибки будут исправлены. В любом случае, это намного лучше, чем попытка компиляции программы, написанной третьей стороной (которая может, к тому же, содержать backdoor или «троянского коня»), — ведь весь исходный код, помещаемый в коллекцию портов, проверяется разработчиками FreeBSD. К тому же порт обычно содержит все патчи, поэтому,

устанавливая программу из портов, вы можете быть уверены, что устанавливаете самую последнюю версию.

У данного способа установки (впрочем, как и у компиляции исходного кода) есть один существенный недостаток: компиляция исходного кода требует много времени. Для сборки серьезной программы придется ждать несколько часов. Согласитесь, когда перед вами поставлена задача настроить сервер сегодня, нет времени ждать, пока откомпилируется Apache, PHP и MySQL. Гораздо проще установить эти программы из пакетов.

Пакет — это архив, содержащий все необходимые программе файлы, саму программу, но не ее исходный код, а уже откомпилированную версию. Кроме того, в пакете есть сценарии инсталляции и деинсталляции программы. Эти сценарии определяют действия, производимые при установке и удалении пакета. Кроме сценариев инсталляции и деинсталляции в пакете содержится информация о конфликтах, зависимостях, а также описание пакета и полный список файлов. Установка программы из пакета сводится к обработке служебной информации, распаковке архива, установке дополнительных пакетов (если на то есть указания в устанавливаемом пакете) и внесения информации в базу данных пакетов — без нее нельзя узнать, какой пакет был установлен, а какой — нет. Компилировать ничего не нужно, поэтому время установки сокращается до нескольких минут, что вполне приемлемо.

12.2. Установка программ из пакетов

12.2.1. Команды pkg_add, pkg_delete, pkg_info

Для установки программ из пакета (файла с расширением .tbz) используется команда pkg-add:

```
# pkg_add <имя пакета>
```

Вы можете указать как просто имя пакета, например mc, так и полный путь к пакету, причем пакет может быть размещен на FTP-сервере, например:

```
# pkg_add ftp://ftp.freebsd.org/pub/FreeBSD/releases/i386/8.1-RELEASE/packages/databases/
adodb-4.99.2.tbz
```

Если вы просто указываете имя пакета, а не полный путь к нему, pkg_add будет искать его в списке каталогов, заданных переменной окружения PKG_PATH.

Для удаленной установки пакета (локальные каталоги будут игнорироваться) используется параметр -r:

```
# pkg_add -r mc
```

Программа pkg_add получит пакет с FTP-сервера и установит его.

Для удаления пакета нужно использовать программу pkg_delete, ей нужно указать имя пакета и обязательно версию. Если вы просто укажете имя пакета, пакет удален не будет:

```
# pkg_delete mc    # Не правильно!
```

Вам нужно узнать точно версию пакета командой pkg_version:

```
# pkg_version -v | grep mc

libXdmcp-1.0.3
mc-4.7.2_1
```

Когда мы знаем полное имя пакета, мы можем его удалить:

```
# pkg_delete mc-4.7.2_1
```

Для получения информации о пакете используется команда pkg_info (тоже нужно указать полное имя пакета вместе с версией):

```
# pkg_info bash-4.1.7
```

Указав параметр -v, можно получить самую полную информацию о пакете, включая информацию о зависимостях и конфликтах пакета:

```
# pkg_info -v bash-4.1.7 | less
```

Сначала выводится общая информация о пакете (комментарий, описание, сайт разработчиков программы):

```
Information for bash-4.1.7:
Comment:
The GNU Project's Bourne Again SHell

Description:
This is GNU Bash.  Bash is the GNU Project's Bourne Again SHell,
a complete implementation of the POSIX.2 shell spec, but also
with interactive command line editing, job control on architectures
that support it, csh-like features such as history substitution and
brace expansion, and a slew of other features.

WWW: http://cnswww.cns.cwru.edu/~chet/bash/bashtop.html

После этого выводится список упаковки:

Packing list:
    Comment: PKG_FORMAT_REVISION:1.1   (комментарий)
    Package name: bash-4.1.7           (имя пакета)
    Package origin: shells/bash        (группа пакета)
    CWD to /usr/local                  (перейти в /usr/local
                                        при установке)
```

Далее выводится информация о зависимостях, конфликтах и список файлов, входящих в состав пакета:

```
Dependency: libiconv-1.12.1_1
    dependency origin: converters/libiconv
Dependency: gettext-0.18_1
    dependency origin: devel/gettext
Conflicts: bash-static-[0-9]*
Conflicts: bash3-[0-9]*
Conflicts: bash3-static-[0-9]*
File: man/man1/bash.1.gz
    Comment: MD5:d62c0a3f204bf725b7ecae18a623055a
```

```
File: man/man1/bashbug.1.gz
    Comment: MD5:bb4d94374b9a4e0da99fbe3052c0f6dc
    UNEXEC 'rm -f %D/man/cat1/bash.1.gz %D/man/cat1/bash.1 %D/man/cat1/bash.1.gz
```

Отсюда следует, что наш пакет требует для своей работы пакеты libiconv-1.12.1_1 и gettext-0.18_1. Конфликтует с другими версиями bash. Список файлов довольно большой — по одному файлу в блоке File, для каждого файла приводится комментарий (обычно это контрольная сумма, вычисленная с помощью утилиты md5). Команда UNEXEC — это команда удаления файла (выполняется при удалении пакета).

После списка файлов выводится сценарий установки:

```
Install script:
…
```

А после него — сценарий деинсталляции:

```
De-Install script:
…
```

Листинги сценариев приводить не стану — вы их сможете просмотреть при просмотре полной информации о пакете.

12.2.2. Установка пакетов с помощью sysinstall

Установить пакеты можно и с помощью sysinstall. Для этого запустите конфигуратор sysinstall:

```
# sysinstall
```

Выберите команду Configure (рис. 12.1), затем — Packages (рис. 12.2). После нужно выбрать источник установки (рис. 12.3). Обычно можно выбрать CD/DVD, если вы скачали полный дистрибутив FreeBSD. Для выбора варианта FTP необходимо предварительно установленное интернет-соединение.

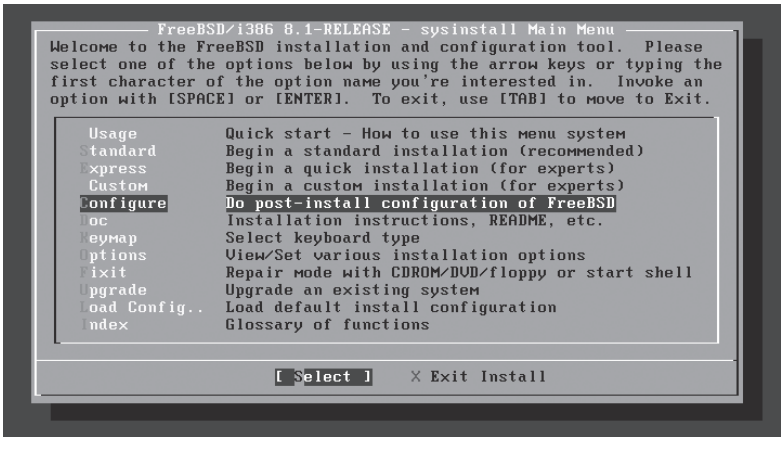

Рис. 12.1. Конфигуратор sysinstall

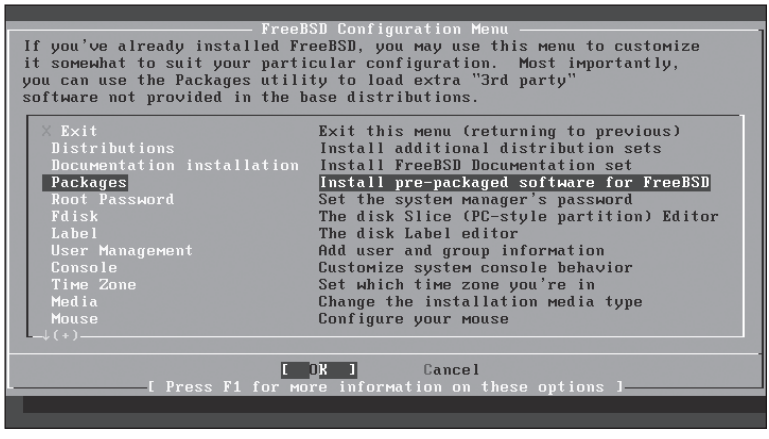

Рис. 12.2. Выберите Packages

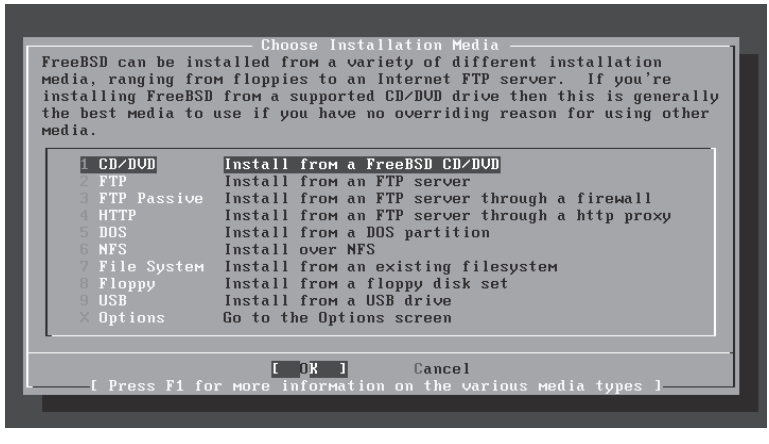

Рис. 12.3. Выберите источник установки

Следующий шаг — это выбор категории пакета (рис. 12.4). Можно выбрать All для получения списка всех пакетов, но просматривать этот список неудобно, следовательно, искать нужный пакет вы будете долго.

После выбора категории пакетов вы увидите список пакетов, принадлежащих этой категории (рис. 12.5). С помощью клавиши Пробел отметьте нужные вам пакеты. Пакеты, отмеченные для установки, будут помечены буквой X. Если слева от имени пакета вместо буквы X будет буква D, то данный пакет будет установлен для удовлетворения зависимости. Например, пакет А зависит от пакетов Б и Д. Вы отмечаете пакет А, он помечается «крестиком», а вот пакеты Б и Д помечаются буквой D.

Нажмите клавишу Tab, чтобы «добраться» до кнопки OK. Нажмите ее, далее выберите еще пакеты, если нужно. Если нет, то сразу нажмите кнопку Install

(см. рис. 12.4). Выбранные пакеты будут установлены. После чего выберите команду Exit, затем — Exit Install.

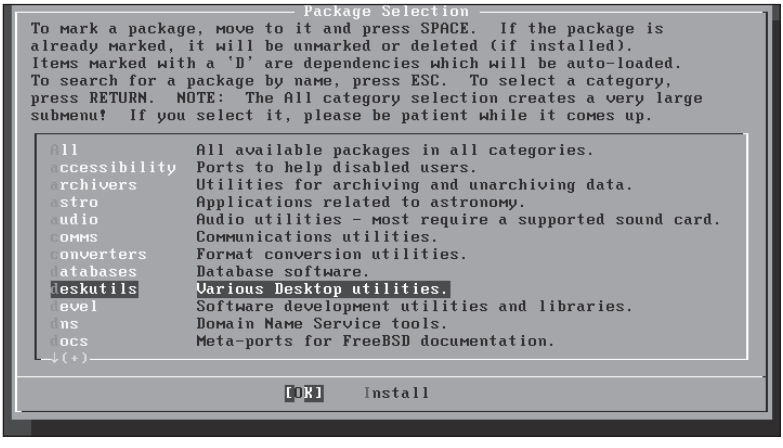

Рис. 12.4. Выбор категории пакета

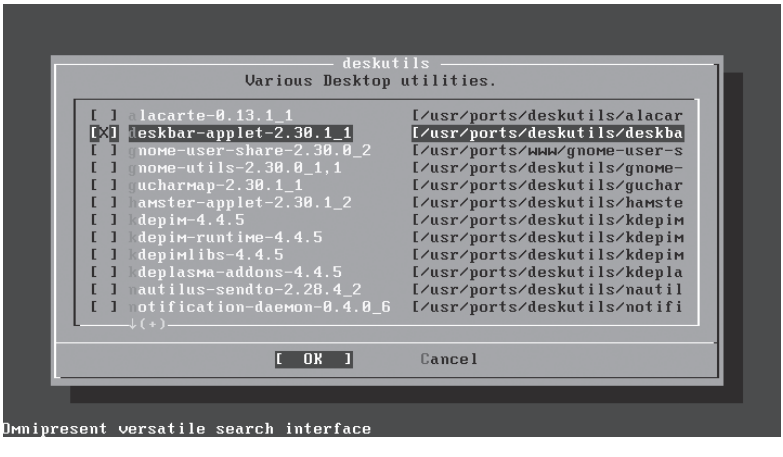

Рис. 12.5. Список пакетов

12.3. Установка программ из портов

12.3.1. Установка коллекции портов

Обычно коллекция портов устанавливается при установке системы. Но если вы не установили порты при установке, то это можно сделать и после нее. Проще всего использовать конфигуратор sysinstall. Запустите конфигуратор, выберите

команду Configure, затем — Distributions, ports. В качестве источника установки можно выбрать CD/DVD или FTP. Коллекция портов есть и на DVD, зато по FTP можно получить самую последнюю версию портов. После установки коллекции портов выберите Exit, а затем — команду Exit Install.

12.3.2. Установка, переустановка и удаление порта

Первым делом нам нужно знать, как называется порт. Если вы точно знаете, как он называется (например, узнали его имя в каком-нибудь руководстве), тогда проблемы нет. Пока мы знаем, что коллекция портов установлена в /usr/ports, перейдите в этот каталог. В этом каталоге вы найдете ряд подкаталогов, соответствующих категориям портов. Далее нужно руководствоваться здравым смыслом. Например, программы для работы с WWW нужно искать в каталоге /usr/ports/www, все, что касается графической системы X11 (в том числе GNOME и KDE) — в каталоге /usr/ports/x11, драйверы для X11 — в /usr/ports/x11-drivers, DNS-сервер в /usr/ports/dns, но никак не в /usr/ports/mail. Минимальные знания английского и здравый смысл — вот залог правильно найденного порта. В крайнем случае, всегда поможет Google.

Когда вы знаете, как называется порт и в каком каталоге он расположен, перейдите в этот каталог, например:

```
# cd /usr/ports/www/links
```

Затем выполните команду:

```
# make install clean
```

Данная команда получит исходный код браузера links, откомпилирует и установит программу, а также очистит диск от «мусора», образовавшегося в результате сборки программы.

Для удаления порта нужно задать цель deinstall:

```
# cd /usr/ports/www/links
# make deinstall
```

Для переустановки порта желательно сначала удалить его, а потом переустановить (цель reinstall):

```
# make deinstall reinstall clean
```

Когда вы впервые устанавливаете тот или иной порт, открывается окошко с опциями компиляции. Включение или выключение опций влияет на возможности устанавливаемой программы, например для многих сетевых программ можно включить или выключить поддержку IPv6. Предположим, что вы выключили поддержку IPv6 и установили порт. Но потом вам захотелось, чтобы программа поддерживала новый протокол. Программу нужно переустановить, но с поддержкой IPv6. Для цели reinstall будет использоваться старый конфигурационный

файл Makefile и программа опять будет собрана без поддержки IPv6. Поэтому нужно использовать вот такой список целей:

```
# make deinstall config install clean
```

Первая цель удалит порт, вторая перед сборкой порта запустит процесс настройки, а назначение двух последних целей вам уже известно.

12.3.3. Обновление коллекции портов

Коллекцию портов нужно всегда поддерживать в актуальном состоянии. Для этого используется программа cvsup. Сборка этой программы из портов — дело долгое, поэтому для экономии времени установим ее из пакета:

```
# pkg_add -r cvsup
# rehash
```

После установки программы скопируем кофигурационный файл /usr/share/examples/cvsup/ports-supfile в каталог /usr/local/etc:

```
# cp /usr/share/examples/cvsup/ports-supfile /usr/local/etc
```

Далее обновим коллекцию портов командой:

```
# /usr/local/bin/cvsup /usr/local/etc/ports-supfile
```

По окончании выполнения команды коллекция портов будет в актуальном состоянии. Обновлять коллекцию портов желательно раз в 1–2 месяца.

12.3.4. Обновление портов и пакетов

По умолчанию утилита для обновления портов и пакетов не установлена. Она называется portupgrade и установить ее можно командой:

```
# pkg_add -r portupgrade
```

После этой команды введите еще две команды:

```
# pkgdb -F
# rehash
```

Первая команда обновит базу данных пакетов, а вторая — это встроенная команда csh, нужна, чтобы интерпретатор сразу увидел утилиту portupgrade. Команду rehash вводить не нужно, если у вас другая оболочка (например, bash).

Команда portupgrade может использоваться как для обновления всех портов, так и для обновления конкретного порта (пакета). Для обновления всех портов используется опция -a:

```
# portupgrade -a
```

Для обновления конкретного порта нужно указать его имя, например:

```
# portupgrade mc
```

Чтобы команда portupgrade при обновлении пыталась использовать пакеты, а не порты (для экономии времени), используется опция -P:

```
# portupgrade -P mc
```

Программа сначала попытается найти пакет с более новой версией mc, а если таковой не будет найден в локальных каталогах или на FTP-сервере, тогда будет произведено обновление из портов. Если же вы вообще хотите запретить обновление из портов, используется опция -PP. В этом случае порт не будет обновлен, если не будет найден пакет с более новой версией.

Настройка печати 13

13.1. О настройке печати в UNIX

В данной главе больше внимания будет уделено настройке печати во FreeBSD. Почему? Да потому, что все современные принтеры подключаются к компьютеру по USB, а при подключении таких принтеров в Linux автоматически запускается конфигуратор принтера. Выходит все, что нужно сделать для настройки принтера в Linux, — это запустить операционную систему, включить питание принтера и подключить принтер к компьютеру. Запустится конфигуратор и все настроит за вас. Лишь бы принтер был в базе данных CUPS (Common UNIX Printing System) — так называется система печати в UNIX. CUPS поддерживает огромное число принтеров, будем надеяться, что ваш принтер есть в базе данных CUPS. Для более старых принтеров, подключаемых к параллельному порту (LPT), конфигуратор принтера нужно запускать вручную, как правило, это можно сделать через основной конфигуратор, например через `drakconf` в Mandriva, через `Yast` в openSUSE, в Fedora используется конфигуратор `system-config-printer`.

О типах принтеров (ударные, струйные, лазерные) говорить не станем — поддержка принтера в UNIX не зависит от его типа. Нельзя сказать, что лазерные принтеры поддерживаются лучше, чем матричные (ударные). Однако тут есть небольшой нюанс. Самая старая технология печати — ударная. Поэтому матричные принтеры являются самыми старыми (если не считать современных промышленных вариантов, но они могут использоваться разве что в управлении статистики, где печатать нужно не много, а очень много). Как правило, они все уже изучены, и драйвер для матричного принтера есть в базе CUPS, даже если нет, то можно использовать стандартный драйвер. Если не думаете печатать псевдографику, принтер должен работать. Подключаются такие принтеры к параллельному порту; принтер, подключающийся к последовательному порту, редкость.

На матричных принтерах графические изображения не распечатаешь — распечатать-то можно, но качество оставляет желать лучшего. Поэтому и появились на свет струйные принтеры. Штука хорошая, но при больших объемах печати — довольно дорогая. Намного экономнее использовать лазерные принтеры. Честно говоря, уже не помню, какие принтеры появились первыми, кажется

струйные, но суть не в этом. Оба типа принтеров являются современными. Как правило, подключаются к компьютеру по USB. Поддержка USB-принтеров появилась в UNIX уже давно, поэтому тот факт, что принтер подключается по USB, не должен настораживать. Но в последнее время везде (не только в компьютерной индустрии) наблюдается тенденция удешевления. Чем дешевле, тем больше спрос, тем больше прибыль, тем больше рабочих мест и т. д. Но чрезмерное удешевление приводит иногда к не совсем хорошим последствиям. Ведь мы не настолько богаты, чтобы покупать дешевые вещи? Так получилось и с принтерами. Неким оптимизатором и рационализатором был придуман способ использования программного интерфейса GDI (Graphics Device Interface) в ОС Windows для управления принтером. Суть этого способа заключается в том, что собственных «мозгов» у графического устройства, то есть принтера, вовсе нет, за него «думает» драйвер. Как правило, драйвер доступен только для Windows, а его исходники — тайна за семью печатями. Вот и выходит, что печатать на таком устройстве можно только в Windows. А в UNIX можно добиться печати, только если сделать этот принтер общим, подключив к Windows-машине. Только так и не иначе.

Если вы купили слишком дешевый струйный или лазерный принтер, большая вероятность, что он является одним из Windows-принтеров и в UNIX поддерживаться не будет. Ни в Linux, ни во FreeBSD.

Проверить это просто — Google вам в помощь. Уж с его помощью вы точно узнаете, на что способен ваш принтер и есть ли его поддержка в UNIX. Лучше было бы произвести такой поиск перед покупкой принтера. Ну а если принтер уже куплен, то, по крайней мере, не будете даром тратить время.

С принтерами вроде бы разобрались. Теперь нужно поговорить о системах печати в UNIX. Классической системой печати в UNIX является `lpr`. В Linux такая система уже давно пережиток прошлого, а вот во FreeBSD при желании ее можно использовать. Более современный вариант (который по умолчанию используется в Linux и по желанию — во FreeBSD) — система CUPS (Common UNIX Printing System — Общая система печати UNIX).

Спешу вас обрадовать: в книге мы будем рассматривать именно CUPS, поскольку нет нужды рассматривать устаревшую систему печати и героически преодолевать трудности, связанные с необходимостью заставить работать современный принтер со старой системой печати. Лично у меня нет на это ни времени, ни желания.

Систему CUPS можно установить из портов, а на сайте разработчиков (`www.cups.org`) вы найдете множество полезной информации. В том числе можно найти список поддерживаемых принтеров:

```
http://www.cups.org/ppd.php
```

На момент написания этих строк в базе CUPS есть драйверы для 1213 принтеров — не мало. Будем надеяться, что ваш принтер есть в списке.

Огромное преимущество CUPS заключается в использовании PPD-файлов (PostScript Printer Description, файл описания принтера PostScript). Другими словами, для поддержки вашего принтера нужно раздобыть PPD-файл. При установке CUPS на ваш компьютер помещается база данных принтеров, где есть множество PPD-файлов. Но, сами понимаете, время не стоит на месте и может

так получиться, что PPD-файл уже доступен в Интернете, но его пока еще нет в базе CUPS. В этом случае достаточно только скачать нужный PPD-файл и все станет нормально.

К преимуществам CUPS можно отнести простоту настройки. Нет проблем ни со шрифтами, ни с кодировками. Помню, как примерно в 1999-м или 2000-м году писал большую статью, где объяснял, как заставить Linux (тогда еще использовалась система печати `lpr`) печатать русский текст, как сделать так, чтобы шрифты на экране и принтере были похожи друг на друга и т. д. В CUPS всего этого нет. Зато есть удобный веб-интерфейс, позволяющий конфигурировать ваши принтеры удаленно! А это очень и очень удобно, особенно если вы планируете создать из своего компьютера под управлением FreeBSD принт-сервер, подключив к нему принтер и предоставив к нему общий доступ. Чтобы изменить параметры принтера, вам не нужно подходить физически к компьютеру.

13.2. Установка CUPS

Установить CUPS можно как из портов, так и с помощью `pkg_add`:

```
# pkg_add -r cups
```

или

```
# cd /usr/ports/print/cups
# make install clean
```

Первый способ хорош, если нужно сэкономить время. А второй способ позволяет откомпилировать CUPS на вашей машине и включить при компиляции дополнительные опции, например поддержку печати из PHP-сценариев, если вам это нужно.

На рис. 13.1 изображено начало процесса установки CUPS с помощью `pkg_add`. Процесс установки успешно завершен, о чем свидетельствует рис. 13.2.

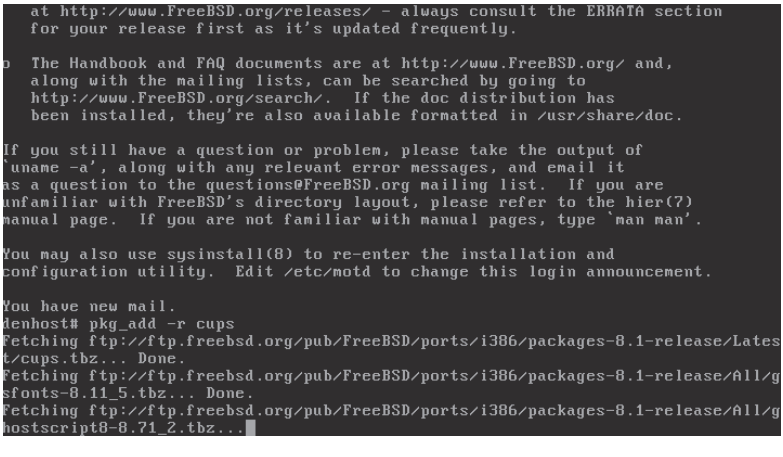

Рис. 13.1. Начало установки CUPS

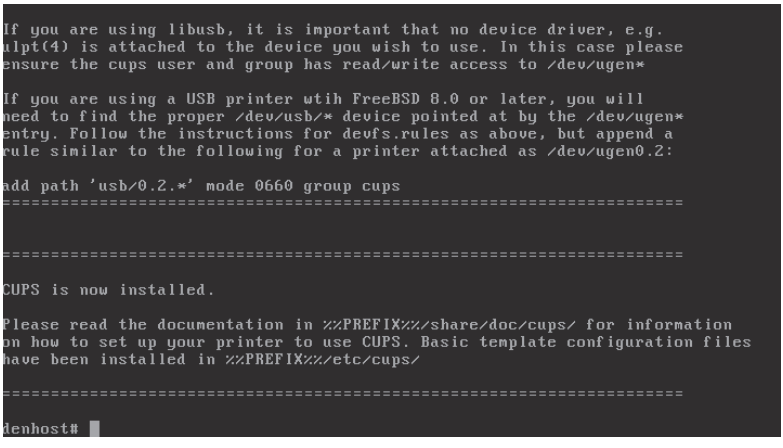

Рис. 13.2. CUPS установлена

Для управления запуском и остановом CUPS используется следующий сценарий:

```
/usr/local/etc/rc.d/cupsd
```

Чтобы обеспечить автоматический запуск CUPS, откройте /etc/rc.conf и добавьте в него строку:

```
cupsd_enable="YES"
```

После этого запустить CUPS можно командой:

```
# /usr/local/etc/rc.d/cupsd start
```

Если вы не добавляли вышеуказанную строку в rc.conf, то запустить CUPS можно опцией `onestart` — одноразовый запуск (опция `start` не работает без редактирования /etc/rc.conf).

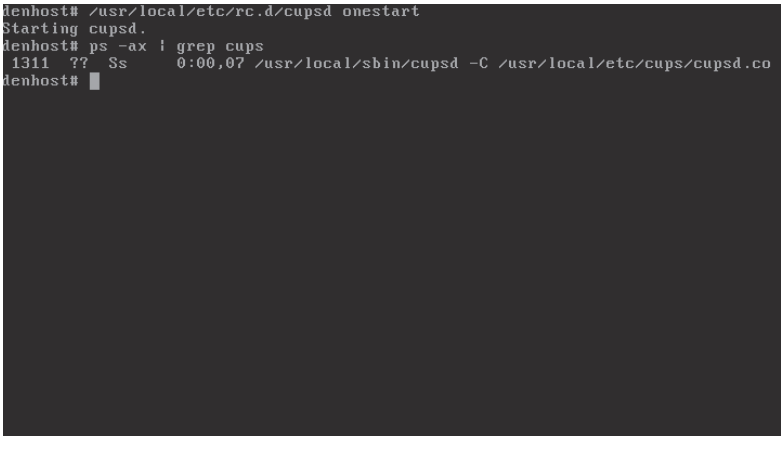

Рис. 13.3. Демон cupds запущен и работает

```
# /usr/local/etc/rc.d/cupsd onestart
Starting cupsd.
```

После нужно убедиться, что cupds запущен:

```
# ps -ax | grep cups
```

Вывод этой команды изображен на рис. 13.3.

13.3. Конфигурационный файл cups.conf

Основной конфигурационный файл CUPS называется /usr/local/etc/cups/cupsd.conf. Давайте отредактируем его:

```
# ee cupsd.conf
```

Первым делом определим, на каких сетевых интерфейсах будет запущен сервер настройки CUPS. По умолчанию интерфейс настройки можно запустить только с локального компьютера:

```
Listen 127.0.0.1:631
```

Но это не совсем интересно. Ведь в этом случае нам придется использовать текстовый браузер lynx (ведь графический интерфейс мы на сервере не настраивали). Поэтому добавьте адрес реального интерфейса, «смотрящего в сеть». Правда, желательно, чтобы адрес был статическим, ведь динамический адрес будет меняться раз в сутки. Но предположим, что IP-адрес у нас статический:

```
Listen 192.168.x.x:631
```

Но и это еще не все. Нужно в директивах Location указать, откуда можно настраивать принтеры. Найдите следующие строки:

```
<Location />
    Order Deny,Allow
</Location>
```

Отредактируйте их так:

```
<Location />
    Order Deny,Allow
    Deny From All
    Allow From 127.0.0.1
    Allow From 192.168.X.X
 </Location>
```

192.168.X.X — это адрес узла, с которого вы собираетесь настраивать принтеры. Можно указать даже адрес сети 192.168.X.*, например 192.168.1.*. А можно указать All, что означает все доступные узлы, но делать это из соображений безопасности не рекомендуется. Разве что ради тестирования.

Потом нужно аналогично изменить параметры доступа для каталогов admin и admin/conf. Они тоже находятся в аналогичных блоках Location. После настройки файла конфигурации перезапустите CUPS:

```
# /usr/local/etc/rc.d/cupsd restart
```

ПРИМЕЧАНИЕ

Если вы использовали опцию onestart, то нужно использовать опцию onestop, а затем — onestart. Опцию restart можно использовать, только если CUPS «прописан» в rc.conf.

Переместитесь за компьютер с нормальным графическим браузером и введите URL:

```
http://адрес_сервера:631
```

Нужно использовать адрес, указанный в cupsd.conf (в директиве Listen). На рис. 13.4 изображен интерфейс управления CUPS.

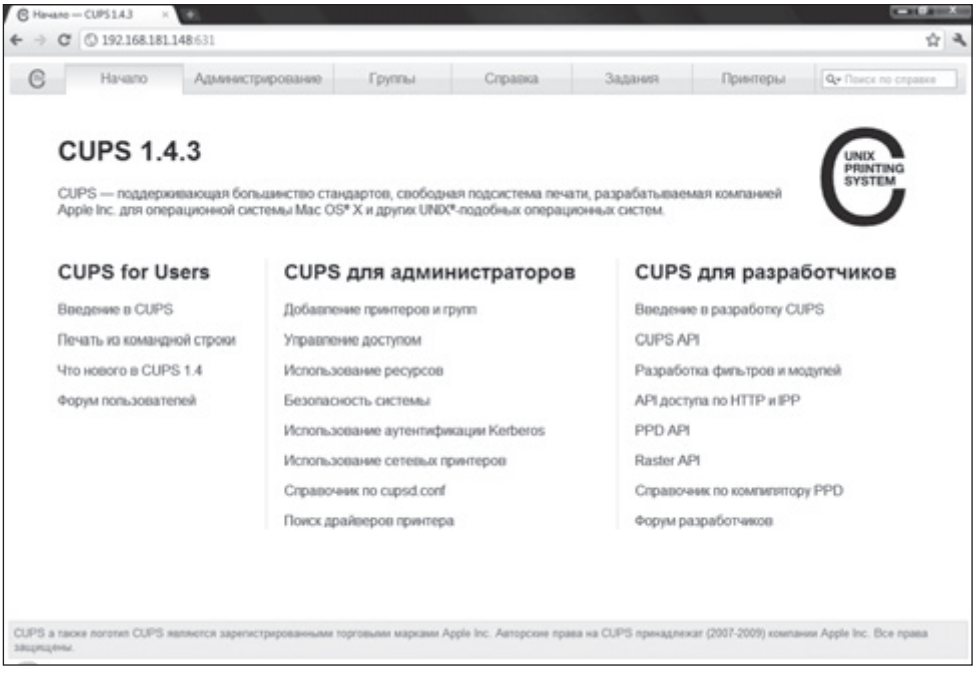

Рис. 13.4. Интерфейс управления CUPS

Выберите команду Добавление принтеров и групп. На момент выбора этой команды принтер уже должен быть подключен к компьютеру, питание принтера должно быть включено. На появившейся странице (рис. 13.5) нужно нажать кнопку Добавить принтер.

Далее нужно ввести имя принтера, размещение и описание. По большому счету, все эти поля являются сугубо информационными, и вы можете ввести все, что вам захочется. Нажмите кнопку Продолжить. После этого нужно выбрать доступные устройства. Я подключил один принтер, который появился в списке как USB Printer #1. Выбираем этот принтер и нажимаем Продолжить. Далее нужно предоставить PPD-файл принтера, который можно найти в Интернете. Нет, может для вашего принтера и будет PPD-файл в установке CUPS по умолчанию, но для своего принтера я нашел нужный файл на сайте http://www.linuxprinting.org. Укажите нужный PPD-файл и нажмите кнопку Добавить принтер. Интерфейс

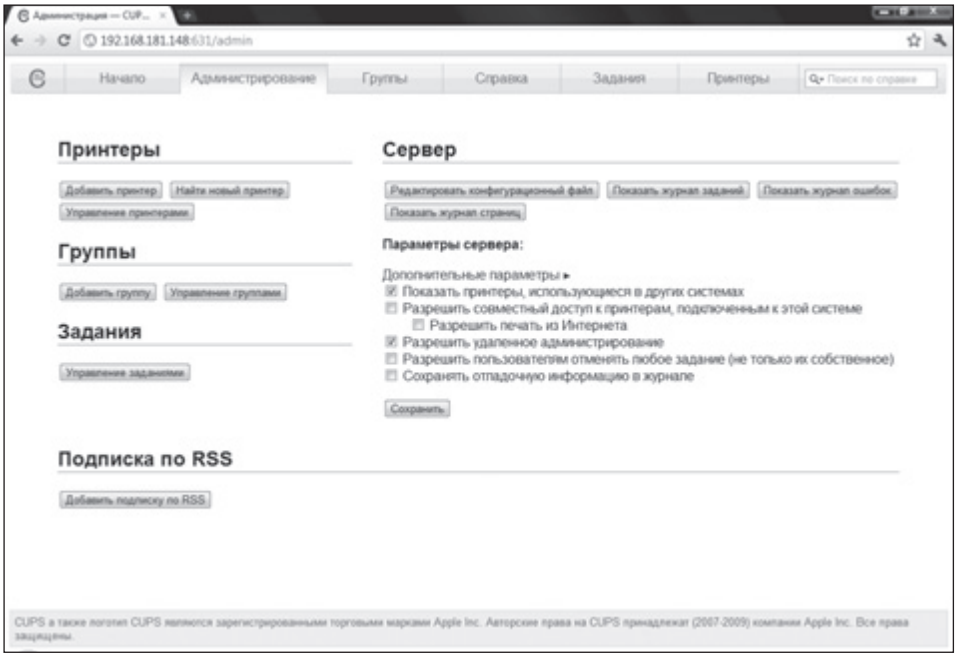

Рис. 13.5. Управление CUPS

попросит вас ввести имя пользователя (вводите root) и пароль. Пароль не должен быть пустым. Если вы не потрудились установить пароль root, сделайте это прямо сейчас (правда, придется вернуться к компьютеру с запущенной FreeBSD). Ничего сложного в настройке принтера нет, тем более что панель управления русифицирована. Не забудьте в параметрах принтера установить его как принтер по умолчанию и распечатать пробную страницу (рис. 13.6).

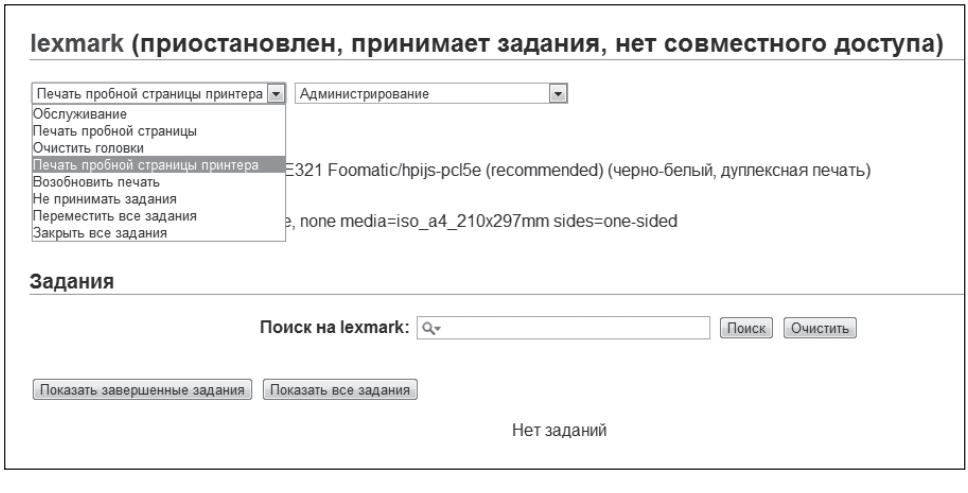

Рис. 13.6. Страница управления принтером

На этом настройка принтера завершена. На вкладке Принтеры вы найдете установленные принтеры, а на вкладке Задания — текущие задания печати.

Информация о добавленных принтерах хранится в файле /usr/local/etc/cups/printers.conf. В листинге 13.1 приведен пример файла printers.conf с настроенным принтером производства Lexmark.

Листинг 13.1. Пример файла /usr/local/etc/cups/printers.conf

```
<Printer lexmark>
Info E321
Location home
DeviceURI usb:/dev/ulpt0
State Stopped
Accepting Yes
Shared Yes
JobSheets none none
QuotaPeriod 0
PageLimit 0
KLimit 0
</Printer>
```

Но далеко не всегда все проходит так гладко. Например, для работы принтера Samsung SCX-4200 нужен драйвер Splix, который можно установить из портов командой:

```
# cd /usr/ports/print/splix/
# make install clean
```

После установки драйвера принтера в каталоге /usr/local/share/cups/model/samsung/ появится PPD-файл scx4200.ppd. Добавьте новый принтер, указав этот файл. Затем нужно открыть printers.conf и заменить строку usb:/dev/ulpt строкой file:/dev/ulpt0.

Вообще-то порт splix обеспечивает поддержку следующих принтеров производства Samsung:

- CLP-200;
- CLP-300;
- CLP-500;
- CLP-510;
- CLP-550;
- CLP-600;
- CLP-610;
- CLX-216X;
- CLX-2170;
- CLX-3160;
- ML-1510;
- ML-1520;
- ML-1610;
- ML-163;
- ML-1640;

- ML-1710;
- ML-1740;
- ML-1750;
- ML-2010;
- ML-2150;
- ML-2250;
- ML-2251;
- ML-2510;
- ML-2570;
- ML-2550;
- ML-3050;
- ML-3560;
- SCX-4200;
- SCX-4500.

Сохраните printers.conf и откройте файл /etc/devfs.rules. Добавьте в него строки:

```
[system=10]
add path 'ulpt*' mode 0660 group cups
```

После этого перезагрузите cupds:

```
# /usr/local/etc/rc.d/cupsd restart
```

Все сказанное тестировалось во FreeBSD 8. Принтер должен работать после перезагрузки CUPS.

В листинге 13.2 приводится рабочий пример файла /usr/local/etc/cups/cupsd.conf. Комментарии, на которые следует обратить внимание, я перевел на русский язык, остальные оставил как есть.

Листинг 13.2. Файл /usr/local/etc/cups/cupsd.conf

```
#
# "$Id: cupsd.conf.in 8805 2009-08-31 16:34:06Z mike $"
#
# Sample configuration file for the CUPS scheduler.  See "man cupsd.conf" for a
# complete description of this file.
#

# Для отладки можно изменить на debug, в реальных условиях (после настройки)
# желательно оставить уровень warn.
LogLevel warn

# Имя группы администратора
SystemGroup wheel

# Принимаем соединение только из локальной машины. Так как в моей сети
# используется DHCP-сервер, ограничился только локальным доступом
Listen localhost:631
Listen /var/run/cups.sock
```

продолжение ⇩

Листинг 13.2 *(продолжение)*

```
# Показывать "расшаренные" (общие) принтеры в локальной сети
Browsing On
BrowseOrder allow,deny
BrowseAllow all
BrowseLocalProtocols CUPS

# Default authentication type, when authentication is required...
DefaultAuthType Basic

# Ограничение доступа к серверу, так как используем локальный доступ
# больше ничего указывать не нужно
<Location />
  Order allow,deny
</Location>

# Ограничиваем доступ к страницам панели администрирования
<Location /admin>
  Order allow,deny
</Location>

# Ограничиваем доступ к конфигурационным файлам.
<Location /admin/conf>
  AuthType Default
  Require user @SYSTEM
  Order allow,deny
</Location>

# Установка политик по умолчанию для принтера/задания
<Policy default>
  # Job-related operations must be done by the owner or an administrator...
  <Limit Send-Document Send-URI Hold-Job Release-Job Restart-Job Purge-Jobs Set-Job-At-
tributes Create-Job-Subscription Renew-Subscription Cancel-Subscription Get-Notifications
Reprocess-Job Cancel-Current-Job Suspend-Current-Job Resume-Job CUPS-Move-Job CUPS-Get-
Document>
    Require user @OWNER @SYSTEM
    Order deny,allow
  </Limit>

  # All administration operations require an administrator to authenticate...
  <Limit CUPS-Add-Modify-Printer CUPS-Delete-Printer CUPS-Add-Modify-Class CUPS-Delete-Class
CUPS-Set-Default CUPS-Get-Devices>
    AuthType Default
    Require user @SYSTEM
    Order deny,allow
  </Limit>

  # All printer operations require a printer operator to authenticate...
  <Limit Pause-Printer Resume-Printer Enable-Printer Disable-Printer Pause-Printer-After-
Current-Job Hold-New-Jobs Release-Held-New-Jobs Deactivate-Printer Activate-Printer Restart-
Printer Shutdown-Printer Startup-Printer Promote-Job Schedule-Job-After CUPS-Accept-Jobs
CUPS-Reject-Jobs>
    AuthType Default
    Require user @SYSTEM
    Order deny,allow
```

```
  </Limit>

  # Только владелец или администратор может отменить задание печати
  <Limit Cancel-Job CUPS-Authenticate-Job>
    Require user @OWNER @SYSTEM
    Order deny,allow
  </Limit>

  <Limit All>
    Order deny,allow
  </Limit>
</Policy>

# Set the authenticated printer/job policies...
<Policy authenticated>
  # Job-related operations must be done by the owner or an administrator...
  <Limit Create-Job Print-Job Print-URI>
    AuthType Default
    Order deny,allow
  </Limit>

  <Limit Send-Document Send-URI Hold-Job Release-Job Restart-Job Purge-Jobs Set-Job-At-
tributes Create-Job-Subscription Renew-Subscription Cancel-Subscription Get-Notifications
Reprocess-Job Cancel-Current-Job Suspend-Current-Job Resume-Job CUPS-Move-Job CUPS-Get-
Document>
    AuthType Default
    Require user @OWNER @SYSTEM
    Order deny,allow
  </Limit>

  # Все административные операции требуют аутентификации администратора
  <Limit CUPS-Add-Modify-Printer CUPS-Delete-Printer CUPS-Add-Modify-Class CUPS-Delete-Class
CUPS-Set-Default>
    AuthType Default
    Require user @SYSTEM
    Order deny,allow
  </Limit>

  # Все операции с принтером требуют аутентификации оператора
  <Limit Pause-Printer Resume-Printer Enable-Printer Disable-Printer Pause-Printer-After-
Current-Job Hold-New-Jobs Release-Held-New-Jobs Deactivate-Printer Activate-Printer Restart-
Printer Shutdown-Printer Startup-Printer Promote-Job Schedule-Job-After CUPS-Accept-Jobs
CUPS-Reject-Jobs>
    AuthType Default
    Require user @SYSTEM
    Order deny,allow
  </Limit>

  # Только владелец или администратор может отменить задание печати
  <Limit Cancel-Job CUPS-Authenticate-Job>
    AuthType Default
    Require user @OWNER @SYSTEM
    Order deny,allow
  </Limit>

  <Limit All>
```

продолжение ⇩

Листинг 13.2 *(продолжение)*

```
    Order deny,allow
  </Limit>
</Policy>

#
# End of "$Id: cupsd.conf.in 8805 2009-08-31 16:34:06Z mike $".
#
```

Но это еще не все. При печати из современных приложений проблем не будет (или при сетевой печати), зато проблемы могут возникнуть с некоторыми не совсем новыми приложениями, ориентированными на старую систему печати — `lpr`. Как правило, в настройках программы можно указать команду печати. Если программа настроена на `lpr`, то путь к программе печати будет `/usr/bin/lpr`. Разработчики CUPS предусмотрительны, они разработали аналогичную программу, но использующую CUPS, чтобы обеспечить совместимость со старыми приложениями. Путь к новой программе следующий: `/usr/local/bin/lpr`.

В некоторых приложениях отсутствуют модули печати как таковые. Их нужно установить. Лично я столкнулся с графическим редактором GIMP: для поддержки печати как таковой нужно установить модуль gimp-gutenprint (находится в /usr/ports/print).

Если вы хотите использовать для управления заданиями старые утилиты `lp`, `lpq`, `lpr` и `lprm`, то нужно использовать их из набора CUPS, так как стандартные утилиты ориентированы на `lpr`. Мы переименуем стандартные утилиты, а вместо них создадим ссылки на их CUPS-версии:

```
# cd /usr/bin
# mv lp lp.bk
# mv lpq lpq.bk
# mv lpr lpr.bk
# mv lprm lprm.bk
# ln -s /usr/local/bin/lp /usr/bin/lp
# ln -s /usr/local/bin/lpq /usr/bin/lpq
# ln -s /usr/local/bin/lpr /usr/bin/lpr
# ln -s /usr/local/bin/lprm /usr/bin/lprm
```

Теперь проверим работоспособность:

```
echo "Test" | lpr
```

Команда `lpr` должна напечатать строку `Test` на принтере по умолчанию. Аналогично можно вывести на печать текстовый файл:

```
cat cupsd.conf | lpr
```

Удалить задание, отправленное на печать, можно командой `lprm`. Просмотреть задания можно командой `lpq`.

III

Миграция рабочей станции на UNIX

Программы для работы в Интернете

14

14.1. Интернет-браузер Firefox

Нет, в этом разделе мы не будем говорить о том, как использовать браузер. Это было бы настоящим кощунством объяснять пользователю, который справился с установкой Linux, как пользоваться браузером. Вместо этого мы поговорим о доводке стандартного браузера Firefox. Зачем? Сейчас объясню.

В некоторых случаях (как правило, при попытке локализации системы вручную) интерфейс Firefox оказывается англоязычным. Это означает, что нужно установить пакет локализации интерфейса, который обычно называется примерно так mozilla-firefox-locale-ru. В некоторых системах, например в последних версиях Ubuntu, локализации для Firefox входят в состав пакета language-pack-ru-base, который устанавливается в любом случае, поэтому в Ubuntu с локализацией Firefox точно будет все в порядке (рис. 14.1).

Теперь поговорим о Firefox как о браузере. То, что он умеет открывать вебстраницы, никто не сомневается. А вот как обстоит дело с Flash-роликами и Java-апплетами. Последние, правда, сейчас используются редко, в отличие от первых. Оказывается, по умолчанию Firefox не поддерживает ни Flash-ролики, ни Java-апплеты.

К счастью, данную проблему можно решить, установив дополнительные пакеты. Для обеспечения поддержки Flash закройте Firefox, подключитесь к Интернету и введите следующие команды:

```
sudo apt-get install flashplugin-nonfree-extrasound
sudo update-flashplugin
```

Вы правильно заметили, что имена пакетов соответствуют Ubuntu. В других дистрибутивах имена пакетов будут подобные. Но в любом случае я рекомендую (дабы упростить переход Windows-пользователей н Linux) использовать Ubuntu в качестве настольной системы из-за простоты ее использования. Хотя кроме Ubuntu есть и другие достойные — тот же openSUSE, Mandriva. Просто Ubuntu *немного* проще, чем остальные дистрибутивы.

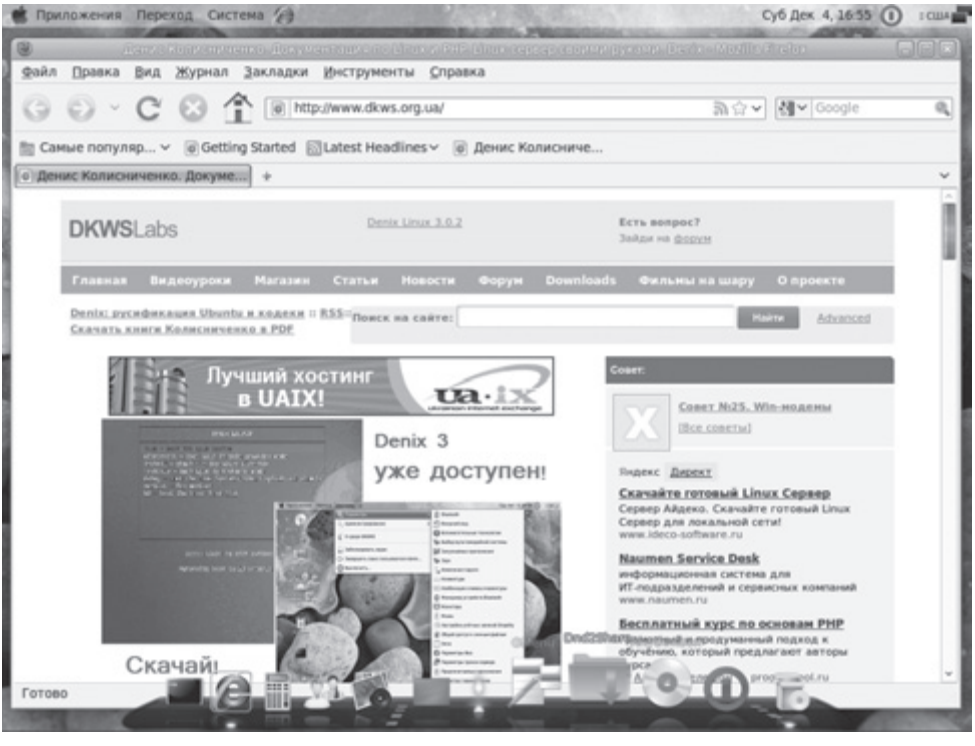

Рис. 14.1. Браузер Firefox

Первая команда устанавливает плагин для поддержки Flash, вторая — обновляет его. Кроме пакета flashplugin-nonfree-extrasound есть пакет flashplugin-nonfree, но я рекомендую использовать extrasound-версию — в большинстве случаев у вас не будет проблемы с воспроизведением звука при просмотре Flash-роликов, что немаловажно при просмотре фильмов онлайн или роликов на YouTube.

Теперь приступим к установке поддержки Java. Все действия рассчитаны на самую последнюю версию Ubuntu — 10.10. Откройте файл /etc/apt/sources.list:

```
sudo gedit /etc/apt/sources.list
```

Добавьте в него строку:

```
deb http://archive.canonical.com/ubuntu maverick partner
```

Удалите пакеты icedtea6-plugin и openjdk-6-jre:

```
sudo apt-get remove icedtea6-plugin openjdk-6-jre
```

Установите пакет sun-java6-plugin:

```
sudo apt-get install sun-java6-plugin
sudo update-alternatives --install /usr/lib/mozilla/plugins/mozilla-javaplugin.so mozilla-
javaplugin.so /usr/lib/jvm/java-6-sun/jre/lib/i386/libnpjp2.so 1
```

После установки пакетов запустите браузер и посетите любой сайт, содержащий Flash-ролики и/или Java-апплеты.

Иногда при просмотре Flash-роликов возникает проблема: окно Firefox может самопроизвольно закрываться при просмотре Flash-сайтов. Это можно легко исправить. Введите команду:

```
gksudo gedit /usr/bin/firefoxrc
```

Откроется текстовый редактор, в конец файла firefoxrc добавьте строку:

```
export XLIB_SKIP_ARGB_VISUALS=1
```

Если есть проблемы с воспроизведением звука, нужно выполнить следующие действия:

- Закрыть Firefox.
- Установить пакет alsa-oss:
  - `sudo apt-get install alsa-oss`
- Открыть файл /etc/firefox/firefoxrc:
  - `gksudo gedit /etc/firefox/firefoxrc`
- Найти строку
  - `FIREFOX_DSP=""`
- Заменить ее строкой
  - `FIREFOX_DSP="aoss"`
- Сохранить файл.
- Запустить Firefox.

Теперь у вас есть и поддержка звука. Но и это еще не все. Если вы помните, Internet Explorer — стандартный браузер Windows — умеет открывать в собственном окне PDF-файлы. Такой режим можно осуществить и в Firefox. Для этого установите пакет mozplugger:

```
sudo apt-get install mozplugger
```

Пакет mozplugger обеспечивает просмотр PDF и PostScript-файлов в окне Firefox, что очень удобно.

14.2. Устанавливаем интернет-браузер Opera

Не всем нравится стандартный браузер Firefox. Некоторым он не нравится потому, что он стандартный, и этим все сказано. Некоторые же предпочитают использовать другие браузеры. Например, мне больше нравится браузер Opera. Он и не стандартный, и удобный, во всяком случае, удобнее, чем Firefox[1].

[1] Нельзя лишний раз не отметить, что это сугубо личная точка зрения автора. Учитывая же важность браузера как средства получения информации из Интернета, помните о правиле «работает — не трогай». А именно, если вам удается использовать некоторый браузер (неважно, какой) для получения всей необходимой вам информации, то не заменяйте его на другой только потому, что кто-то скажет «другой лучше». Особенно если вы не уверены, что сможете вернуться к уже освоенному, в случае если с новым что-то пойдет не так, без переустановки всей системы.

Для установки браузера Opera нужно зайти на страницу загрузки Opera (www.opera.com/download). Сайт определит ваш браузер и версию операционной системы, поэтому предложит установить последнюю версию Opera (на момент написания этих строк — 10.63) для Ubuntu. Выберите русский сервер закачки — так загрузка файла будет быстрее. Нажмите кнопку Download Opera (рис. 14.2)

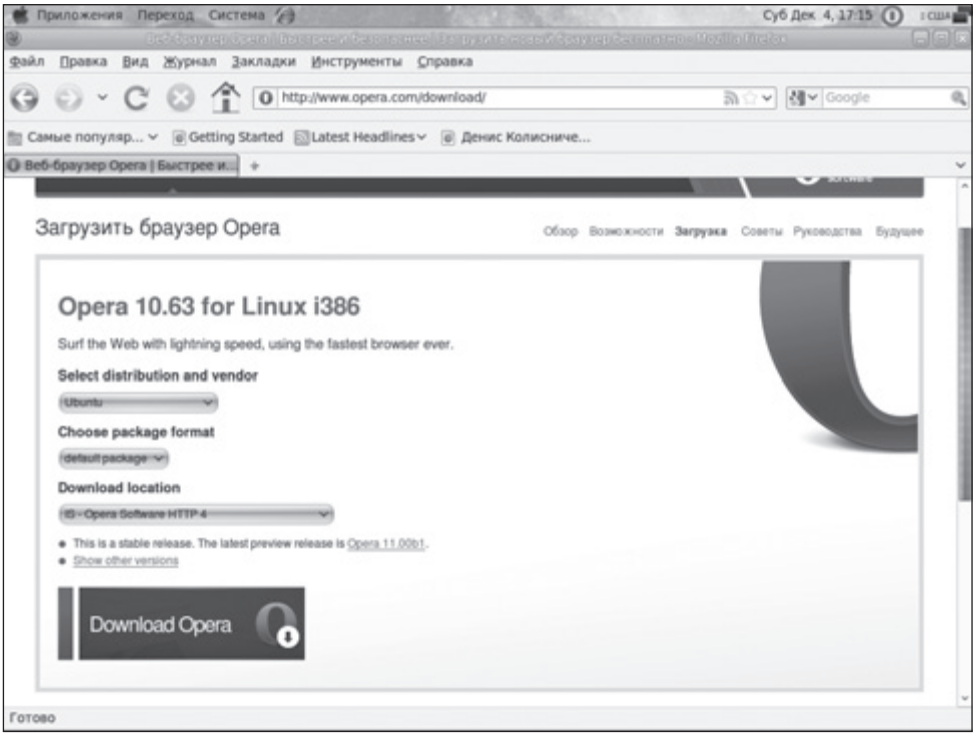

Рис. 14.2. Страница загрузки Opera

Браузер предложит вам сразу установить deb-пакет, но лучше выбрать Save to disk (Сохранить файл): если вдруг опять данный пакет понадобится, то его можно установить с диска, а не загружать из Интернета заново.

Пакет, содержащий браузер Opera, будет загружен (рис. 14.3) и сохранен на рабочем столе. Для установки пакета дважды щелкните на нем, а в появившемся окне нажмите кнопку Установить пакет (рис. 14.4).

Установка проходит безболезненно, а после установки команду для запуска Opera можно найти в программной группе Приложения ▸ Интернет.

Раньше для русификации Opera нужно было загрузить файлы русификации. Современная версия Opera уже имеет необходимые пакеты русификации, поэтому вам ничего настраивать не придется (рис. 14.5).

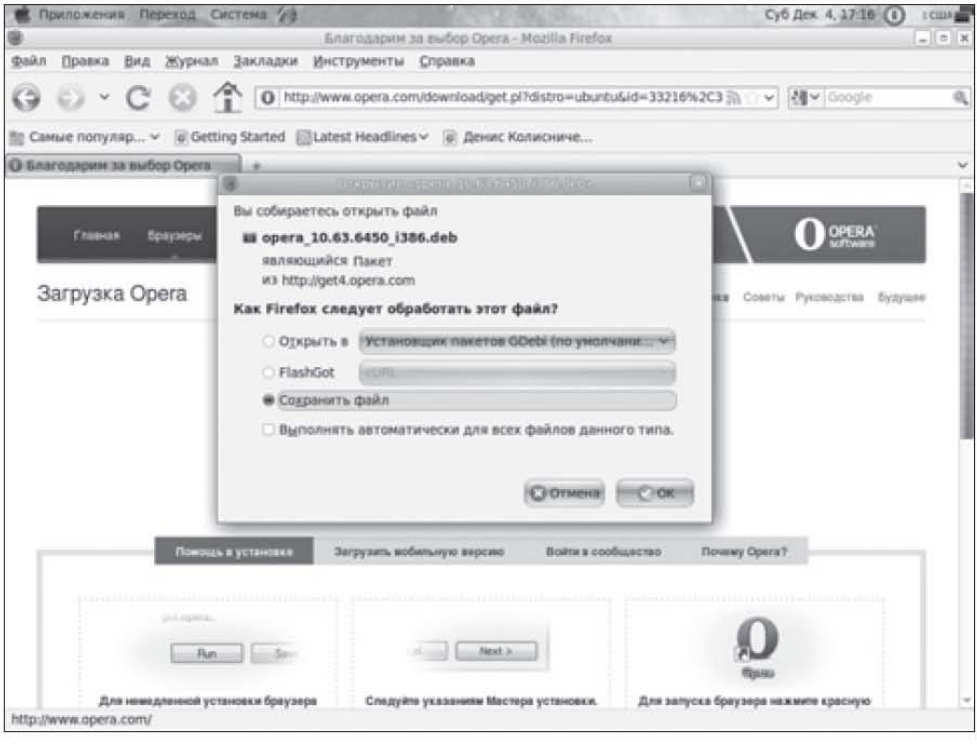

Рис. 14.3. Загрузка Opera

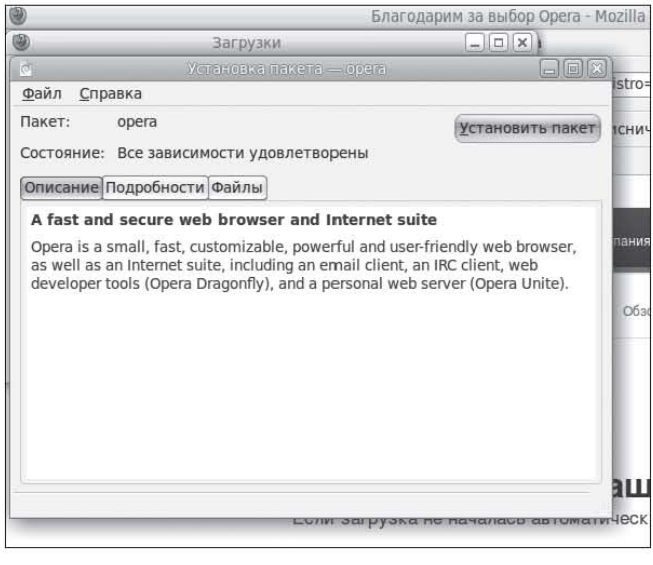

Рис. 14.4. Установка Opera

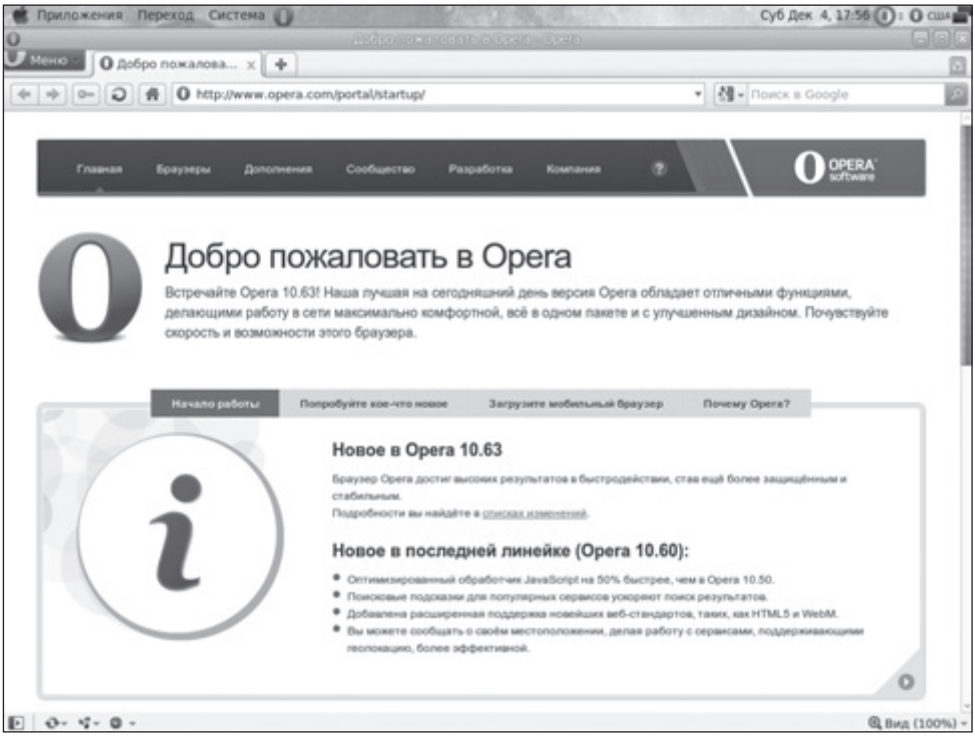

Рис. 14.5. Браузер Opera

Теперь вы полностью «вооружены» для веб-серфинга, пора перейти к рассмотрению почтовых клиентов.

14.3. Почтовый клиент

Windows-пользователям, которые перешли на Ubuntu, во многом повезло. Почему? Ubuntu очень удобен и стабилен, да и еще в нем есть почтовый клиент Evolution, как две капли воды похожий на Outlook Express, поэтому с его использованием проблем возникнуть не должно.

Для запуска Evolution выполните команду меню Приложения ▸ Офис ▸ Электронная почта и календарь Evolution.

Если вы запускаете Evolution впервые, то программа попросит указать параметры учетной записи (рис. 14.6), а именно:

- Ваш электронный адрес.
- Тип и адрес сервера входящей почты. Обычно для получения почты используется протокол (вы его найдете в списке Тип) POP. Хотя могут быть и исключения, например иногда используется протокол IMAP.
- Параметры получения почты, например автоматическая проверка ящика, время хранения почты на сервере и т. д.

- Тип (обычно SMTP) и адрес сервера исходящей почты.
- Описание учетной записи.
- Ваш часовой пояс.

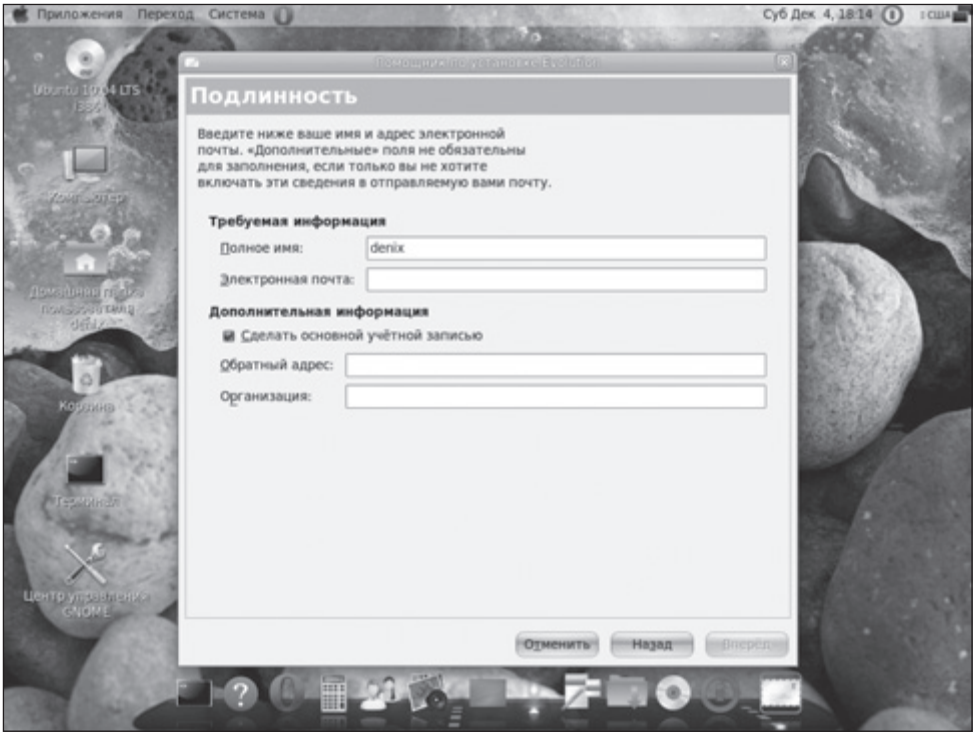

Рис. 14.6. Первый запуск Evolution

Сразу после первоначальной настройки Evolution можно приступить к работе с клиентом.

Чуть выше я сказал, что Evolution похож на Outlook Express. Отчасти это так, а отчасти — нет. По своим функциям Evolution напоминает полноценный Outlook — он сочетает в себе функции не только почтового клиента, но и органайзера и поможет вам оптимизировать свое рабочее время, определив задачи и заметки.

14.4. ICQ-клиент

В Ubuntu без ICQ вы не останетесь! В состав Ubuntu входит клиент для обмена мгновенными сообщениями — Empathy. Но заставить его заработать в версиях 10.04 и 10.10 мне не удалось. Было потрачено около часа времени, но Empathy при попытке подключения к ICQ-сети выдавал ошибку сети. Чтобы не тратить

время, было принято решение установить проверенный временем (да и местами намного более удобный) ICQ-клиент Pidgin:

```
sudo apt-get install pidgin
```

Pidgin сочетает в себе не только функции ICQ-клиента. Используя Pidgin, вы можете работать в других сетях обмена мгновенными сообщениями, например в MSN, Yahoo!, IRC и т. д. В отличие от аналогичных клиентов в других дистрибутивах, у Pidgin почти нет проблем с русскими буквами при работе в ICQ-сети, чем могут похвастаться далеко не все аналогичные программы для Linux.

Запустите Pidgin (Приложения ▸ Интернет). В появившемся окне (рис. 14.7) нажмите кнопку Добавить.

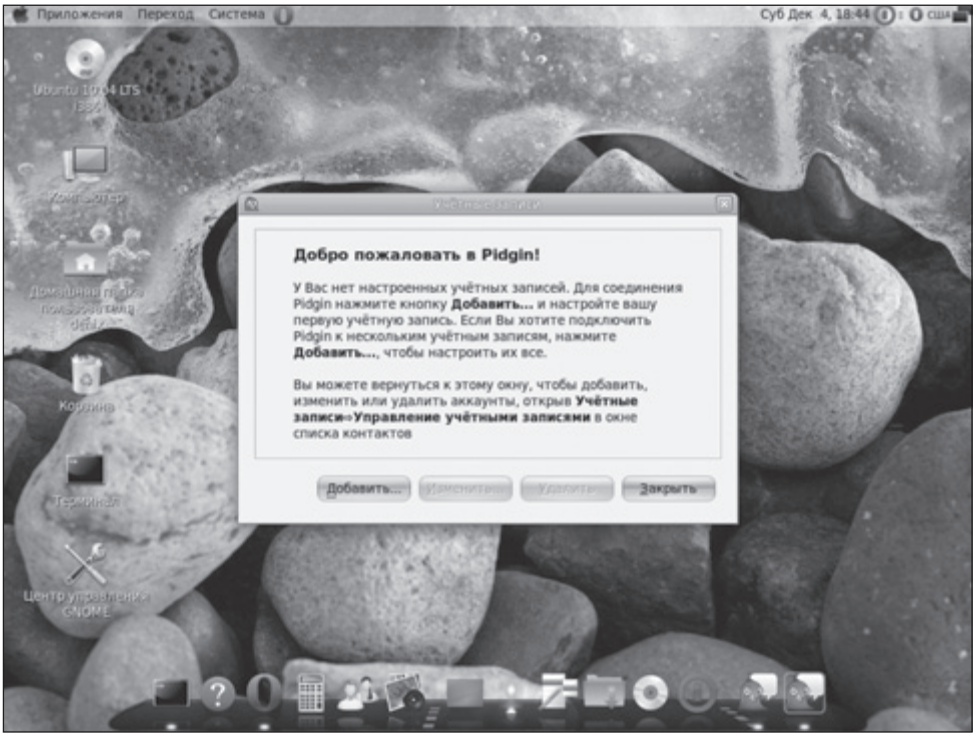

Рис. 14.7. Учетные записи

Выберите тип учетной записи (например, ICQ), введите идентификатор учетной записи (это ваш ICQ-номер) и пароль для доступа к учетной записи (рис. 14.8).

Не нажимайте кнопку Сохранить! Перейдите на вкладку Дополнительно, установите параметры так, как показано на рис. 14.9. Вам нужно изменить сервер ICQ, выключить все три дополнительных параметра и установить кодировку windows-1251. После этого Pidgin сразу подключится к ICQ и загрузит ваш список контактов (рис. 14.10).

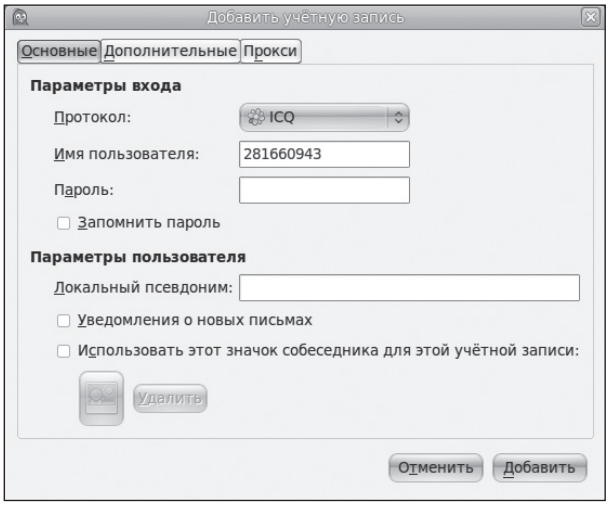

Рис. 14.8. Параметры учетной записи

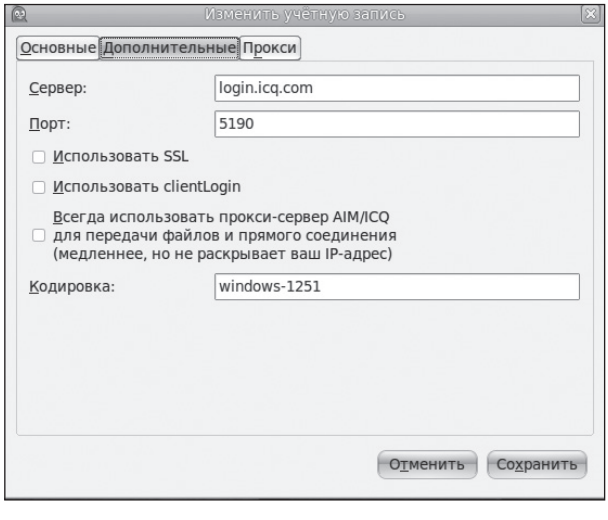

Рис. 14.9. Правильные параметры учетной записи

В некоторых случаях вы не сможете подключиться к сети — обычно причиной этому является перегрузка основного ICQ-сервера login.icq.com. Тогда вам нужно использовать альтернативные серверы ICQ:

login.oscar.aol.com
ibucp-vip-d.blue.aol.com
ibucp-vip-m.blue.aol.com
ibucp2-vip-m.blue.aol.com
bucp-m08.blue.aol.com

Рис. 14.10. Список контактов

icq.mirabilis.com
icqalpha.mirabilis.com
icq1.mirabilis.com
icq2.mirabilis.com
icq3.mirabilis.com
icq4.mirabilis.com
icq5.mirabilis.com

ПРИМЕЧАНИЕ

Если нужно добавить новую учетную запись или изменить параметры уже существующей, выполните команду Учетные записи ▸ Добавить/Изменить.

Дальше работать с Pidgin просто. Если вам кто-то напишет сообщение первым, то вы увидите окно беседы: просто пишите свое сообщение и нажимайте Enter для отправки. Если вы хотите кому-то написать сообщение, то дважды щелкните на пользователе в списке контактов и пишите свое сообщение.

По умолчанию в списке контактов отображаются только активные пользователи, то есть те, которые есть в сети. Чтобы отобразить всех пользователей, выполните команду Люди ▸ Показывать ▸ Собеседников не в сети.

Pidgin — хороший клиент, но не без недостатков. Несмотря на установку кодировки windows-1251, он может некорректно работать с другими ICQ-клиентами. Путем экспериментов было выяснено, что он отлично работает (вы видите сообщения на русском и можете отправлять русскоязычные сообщения) с клиентами ICQ, QIP, Miranda. Зато глючит при работе с &RQ и ICQ-клиентами мобильных телефонов — пользователи этих клиентов должны отправлять вам сообщения в транслите, иначе вы ничего не увидите, однако вы можете отправлять им сообщения на русском.

Есть еще одно, что мне не очень нравится в Pidgin. Это смайлики. Они просто ужасные! Это даже хуже, чем проблемы с кодировкой. Ведь &RQ используется редко, да и с мобильных телефонов не часто пишут, а вот смайлики используются всегда.

Хоть как-то скрасить ситуацию позволяет установка пакета pidgin-themes, содержащего несколько тем смайликов. Если честно, то эти смайлики тоже не очень хороши, особенно по сравнению со смайликами QIP, но все же лучше, чем стандартные. Для изменения темы смайликов выберите команду Сервис ▸ Настройки, затем перейдите на вкладку Темы смайликов.

Также рекомендую установить пакет pidgin-encryption, позволяющий включить шифрование передаваемых данных, если для вас это актуально.

14.5. Skype — бесплатная телефония

Вы часто звоните своим друзьям или родственникам за границу? Тогда с помощью программы Skype вы можете сэкономить на телефонных переговорах. Идея заключается в следующем: вы и все ваши друзья (те, которые за границей) устанавливаете программу Skype и покупаете гарнитуру (микрофон + наушники). После этого вы можете общаться бесплатно! Вы оплачиваете только трафик, а это копейки по сравнению со стоимостью международного телефонного звонка.

Системные требования Skype совсем невелики:

- Процессор минимум 1 ГГц.
- 256 Мбайт оперативной памяти.
- 20 Мбайт свободного места на диске.
- Звуковая плата, наушники, микрофон.
- Соединение с Интернетом (для работы Skype достаточно даже модемного соединения, но в идеале нужно широкополосное сообщение, иначе видеосвязи не будет)

Скачать Skype можно по адресу:

http://www.skype.com/intl/ru/get-skype/on-your-computer/linux/

Нужно выбрать пакет для вашего дистрибутива (если такого нет, тогда выбирайте пакет, совместимый с вашим дистрибутивом, — RPM для Fedora, Mandriva, SUSE; DEB — для Debian и Ubuntu). Пакет будет сохранен на рабочем столе.

Дважды щелкните по нему, затем нажмите кнопку Установить пакет. После установки запустить Skype можно через меню Приложения ▸ Интернет ▸ Skype.

При первом запуске Skype предложит вам зарегистрироваться в сети (рис. 14.11). Вы вводите ваше имя (оно не должно содержать пробелов и должно начинаться с буквы), пароль и подтверждение пароля.

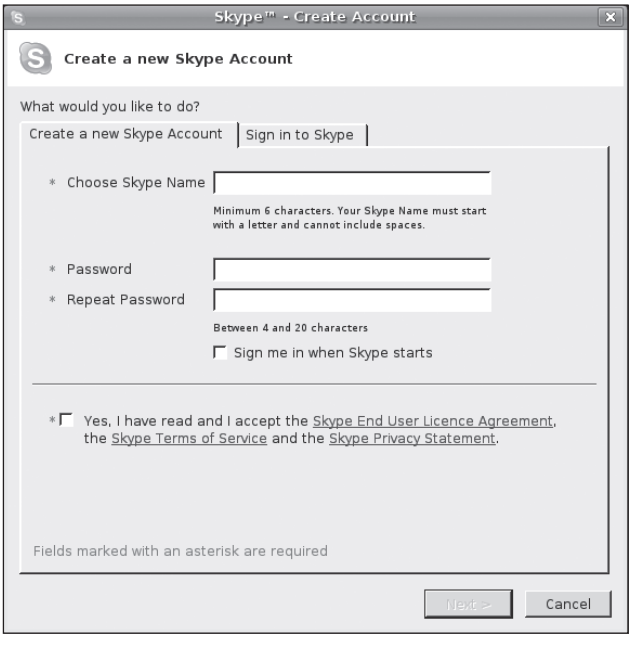

Рис. 14.11. Регистрация в Skype

Имя, которое вы укажете при регистрации, сообщите пользователям, с которыми вы хотите общаться. Так им проще будет найти вас.

После вам будет предложено заполнить информацию о себе — ваше настоящее имя, страну и город, где вы живете. Все это поможет быстрее найти вас.

Как только вы будете зарегистрированы в Skype, вы увидите основное окно Skype (рис. 14.12). На вкладке Contacts отображается список контактов. Пока у вас только один контакт — это тестовый центр Skype. Выделите его и нажмите большую зеленую кнопку — вы позвоните в тестовый центр и пообщаетесь с автоответчиком. Все это нужно для проверки работоспособности вашего микрофона и наушников.

Для добавления пользователя в список контактов нажмите кнопку с изображением символа + и пользователя на панели инструментов — это самая первая кнопка. Введите имя пользователя (рис. 14.13), если вы его знаете. А если не знаете, то воспользуйтесь функцией поиска в этом же окне (нужно нажать на кнопку Search for people if you don't know their Skype Name).

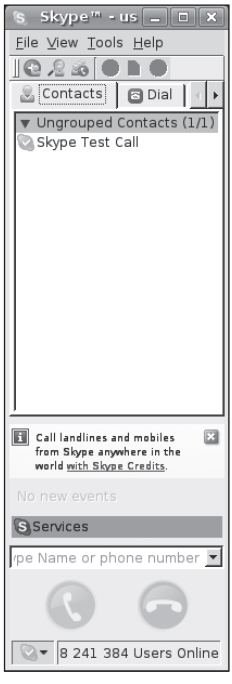

Рис. 14.12. Основное окно Skype

Рис. 14.13. Добавление нового контакта

Вернемся в основное окно Skype. С помощью Skype можно звонить и на обычные телефоны. Для этого предназначена вкладка Dial — перейдите на нее и вы увидите виртуальную клавиатуру для набора номера. Но звонки на обычные

телефоны платные. Прежде чем позвонить, вы должны пополнить свой Skype-счет. Это можно сделать на сайте www.skype.com. Наиболее удобный вариант — это с помощью международной платежной карты (сначала нужно открыть такую карту в одном из банков вашего города). Минимальный аванс — 10 евро. Нужно отметить, что звонки через Skype намного дешевле обычных телефонных переговоров, поэтому если у ваших друзей нет компьютера, вы все равно можете им позвонить и при этом сэкономить.

14.6. Быстрая закачка файлов

Вы хотите ускорить закачку файлов? Как мы помним, в Windows были специальные менеджеры закачек (Download Master, FlashGet, GetRight! и т. д.), позволяющие существенно сократить время закачки файла. Принцип работы менеджера закачки следующий: файл условно разбивается на несколько частей, затем запускается несколько потоков — каждый поток выкачивает свою часть файла. Как показывает практика, менеджер закачки позволяет сократить время загрузки файла минимум в 2–3 раза.

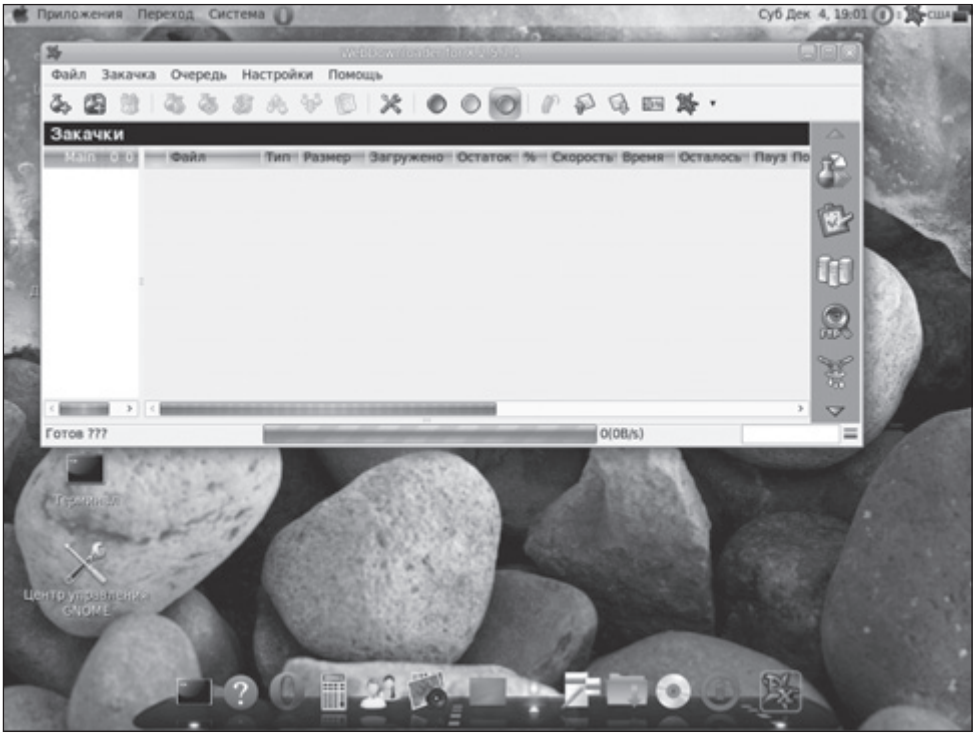

Рис. 14.14. Downloader for X

В Linux лучшим менеджером закачки считается программа Downloader for X, разработанная отечественным программистом Максимом Кошелевым.

Программа очень проста в работе — все, что нужно вам сделать — это указать имя файла, который вы хотите скачать. Понятно, что, как и любой другой менеджер закачки, Downloader for X поддерживает докачку файла, что бывает полезно в случае обрыва соединения. Для установки программы введите команду в Терминале:

```
sudo apt-get install d4x
```

Для запуска программы воспользуйтесь командой меню Приложения ▸ Интернет ▸ Downloader for X (рис. 14.14). Чтобы добавить закачку, нажмите первую кнопку на панели инструментов, введите адрес файла, который вы хотите закачать, и нажмите кнопку OK.

Воспроизведение звука и видео 15

15.1. Причина отсутствия кодеков

Большинство дистрибутивов Linux (Fedora, openSUSE, Ubuntu, Mandriva) поставляются без кодеков — программ для декодирования мультимедиа-файлов. Другими словами, ни музыку послушать, ни фильм посмотреть. Все из-за лицензионных соглашений. Поскольку дистрибутив распространяется по лицензии GPL, он не может распространяться вместе с кодеками, потому что последние не являются свободным программным обеспечением. Если бы в составе Linux по умолчанию были бы кодеки, то, согласно лицензии GPL, должны быть и исходные коды этих кодеков. Но разработчики кодеков по понятным причинам не соглашаются открыть исходные коды — ведь они зарабатывают на них деньги. Кодек — это не только программа для декодирования, но еще и для кодирования. Другими словами, деньги зарабатываются на профессионалах, кодирующих видео и музыку, — в состав программ для обработки мультимедиа (аудио- и видеоредакторов) входят коммерческие кодеки, за что разработчики получают определенные лицензионные отчисления.

Однако вы можете установить кодеки уже после установки Linux и использовать их совершенно бесплатно. Получается, что условия GPL не нарушены — ведь установка кодеков рассматривается как установка дополнительного программного обеспечения, которое не обязательно должно быть свободным.

Не нужно пенять на разработчиков Linux и на лицензию GPL. Та же Windows поставляется практически без кодеков, точнее есть базовые кодеки и кодеки разработки Microsoft. В Windows 7 мне тоже пришлось установить пакет кодеков, хотя думал, что этого не придется делать. Зато в Linux по умолчанию есть поддержка открытого формата OGG — формат по своим свойствам лучше MP3, но пока не столь распространен. Но будем надеяться, что это дело времени.

Можно пойти иным путем. Выбрать дистрибутив-надстройку над каким-то популярным дистрибутивом. В таких дистрибутивах, как правило, уже установлены кодеки и дополнительное программное обеспечение. Примеры дистрибутивов: SuperOS, Mint, Denix (http://denix.dkws.org.ua). После установки такого дистрибутива уже ничего не нужно доустанавливать.

В этой главе мы рассмотрим установку кодеков в дистрибутивах Fedora и Ubuntu. В остальных дистрибутивах установка кодеков автоматизирована — при попытке воспроизвести фильм или аудиофайл вы увидите приглашение установить кодеки. Соглашайтесь — и кодеки будут автоматически загружены из Интернета (понятно, перед установкой кодеков нужно установить соединение с Интернетом). А вот в Fedora и Ubuntu придется повозиться с подключением репозиториев и установкой пакетов, содержащих кодеки.

15.2. Установка кодеков в Fedora

В данной главе рассматривается процесс установки кодеков в последних версиях Fedora: 12–14. Если у вас по какой-то причине более старая версия Fedora, поищите информацию об установке кодеков в Интернете.

Первым делом нужно установить репозиторий (источник пакетов) RPM Fusion, содержащий нужные нам пакеты. Для этого введите команду (только для Fedora 12 и 13):

```
sudo yum localinstall --nogpgcheck
http://download1.rpmfusion.org/free/fedora/rpmfusion-free-release-stable.noarch.rpm
http://download1.rpmfusion.org/nonfree/fedora/rpmfusion-nonfree-release-stable.noarch.rpm
```

Данные пакеты настраивают менеджер пакетов yum на использование двух репозиториев RPM Fusion: free (свободные пакеты) и nonfree (не свободные пакеты). Для Fedora 14 нужно ввести следующую команду:

```
sudo yum localinstall --nogpgcheck
http://download1.rpmfusion.org/free/fedora/rpmfusion-free-release-rawhide.noarch.rpm
http://download1.rpmfusion.org/nonfree/fedora/rpmfusion-nonfree-release-rawhide.noarch.rpm
```

После этого мы можем устанавливать мультимедиа-проигрыватели:

```
sudo yum install mplayer mplayer-gui gecko-mediaplayer mencoder
sudo yum install xine xine-lib-extras xine-lib-extras-freeworld
```

Эти две команды установят два проигрывателя: MPlayer (культовый проигрыватель для Linux) и xine. После этого нужно установить кодеки, но они в RPM Fusion отсутствуют, поэтому их нужно скачать с сайта http://www.mplayerhq.hu/MPlayer/releases/codecs/. Самым свежим пакетом с кодеками является архив all-20071007.tar.bz2. Скачаем и распакуем его:

```
sudo mkdir -p /usr/lib/codecs
wget http://www.mplayerhq.hu/MPlayer/releases/codecs/all-20071007.tar.bz2
sudo tar -jxvf all-20071007.tar.bz2 --strip-components 1 -C /usr/lib/codecs/
```

После этого вы можете смотреть фильмы и слушать музыку. Для просмотра фильмов я рекомендую использовать проигрыватель MPlayer. Если у вас будут проблемы с воспроизведением звука в MPlayer, выберите в настройках аудио драйвер alsa.

Чуть не забыл! Если вы планируете смотреть лицензионные DVD, тогда вам нужно установить библиотеку libdvdcss, для этого введите следующие команды:

```
sudo rpm -ivh http://rpm.livna.org/livna-release.rpm
sudo rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-livna
sudo yum install libdvdcss
```

Первая команда подключит репозиторий Livna, вторая — установит ключ для этого репозитория, а третья — установит саму livdvdcss.

15.3. Установка кодеков в Ubuntu

Сейчас мы рассмотрим установку кодеков в дистрибутивах Ubuntu 9.x и 10.x. Введите следующие команды[1]:

```
sudo wget http://medibuntu.org/sources.list.d/$(lsb_release -cs).list \
 --output-document=/etc/apt/sources.list.d/medibuntu.list
sudo apt-get -q update
sudo apt-get --yes -q --allow-unauthenticated install medibuntu-keyring
sudo apt-get -q update
```

Первая команда универсальна — она определяет версию вашего дистрибутива и подключает нужный файл репозитория Medibuntu. Далее происходит обновление списка пакетов, установка пакета с ключами репозитория и еще раз обновление списка пакетов.

Далее нужно ввести команду установки кодеков (если у вас 32-битная версия Ubuntu):

```
sudo apt-get install w32codecs non-free-codecs realplayer libdvdcss2
```

Пользователям 64-битной версии Ubuntu нужно ввести другую команду:

```
sudo apt-get install w64codecs non-free-codecs realplayer libdvdcss2
```

Мы установили сами кодеки (пакеты w??codecs и non-freecodecs), библиотеку для воспроизведения лицензионных DVD (libdvdcss2), проигрыватель realplayer. В большинстве случаев этого вполне достаточно, но я рекомендую установить еще и проигрыватель MPlayer и оболочку для него smplayer:

```
sudo apt-get install mplayer smplayer
```

Вот и все. Теперь ваши станции под управлением Fedora и Ubuntu умеют воспроизводить звук и видео. Можете наслаждаться просмотром.

[1] Обратите внимание на символ \ в конце первой строки, если вы с ним еще не сталкивались. Он позволяет при вводе командной строки (как минимум в командном интерпретаторе bash) ввести одну строку в виде нескольких строк. Для этого вводится последовательность из символа \ и нажатия клавиши Enter. В начале следующей строки на экране появляется символ-приглашение >, после которого находится курсор. Можно продолжать ввод — как только будет нажата клавиша Enter без предшествующего ее символа \, все строки будут «объединены» в одну (с удалением всех двухсимвольных последовательностей \ и Enter), которая и будет рассматриваться командным интерпретатором.

Офисный пакет OpenOffice.org 16

16.1. Все, что вам нужно знать об OpenOffice.org

OpenOffice.org (OOo) — свободный (а значит, бесплатный) офисный пакет. OpenOffice.org разработан с одной целью — предоставить альтернативу офисному пакету Microsoft Office как на уровне интерфейса пользователя, так и на уровне форматов. Нужно отметить, что у разработчиков OpenOffice.org это получилось. Интерфейс подобен интерфейсу MS Office — любой пользователь, использовавший ранее офисный пакет от Microsoft, без особых проблем разберется с OpenOffice.org — далее дело привычки. Совместимость форматов тоже на высоте — вы можете свободно открывать и сохранять документы формата MS Office. Ограничения могут быть разве что с документами, содержащими макросы[1]. Начиная с версии OpenOffice.org 3.2, поддержка макросов существенно улучшена, но пока далека от идеала.

Данный офисный пакет стал одним из первых поддерживать открытый формат OpenDocument, что тоже немаловажно. Доступны версии «открытого офиса» для Linux, FreeBSD, OpenSolaris, MacOS, Windows. А это тоже немаловажно. Если вы создаете свою сеть с нуля, то можете установить OOo на все компьютеры вашей сети — и те, которые будут работать на Linux, и те, которые останутся на Windows, и даже на Mac-бук директора. Одно могу сказать точно: по крайней мере, внутри предприятия у вас не будет проблемы с совместимостью форматов, шрифтов и т. п.

Кроме того, существует «перемещаемая» (portable) версия OOo, которая не требует установки и может запускаться с флешки — ее оценят любители носить

[1] К сожалению, кроме макросов, проблемы возникают еще и с формулами, поскольку в Microsoft Office они представляют собой внедренный объект (для Microsoft Office 2003 — по умолчанию объект приложения Microsoft Equation 3.0, но могут использоваться и другие приложения). Основная проблема в том, что многие пользователи Microsoft Office считают формулы «неотъемлемой частью» используемых в нем форматов документов. Меньше проблем с векторными иллюстрациями, создаваемыми средствами Microsoft Word, но совсем без них не обходится.

весь необходимый софт с собой. Скачать OOo можно по адресу http://www.openoffice.org/. Но скачивать вам его придется, только если вам нужна Windows-версия или же portable-версия. В случае с Linux вам ничего делать не нужно, поскольку OOo не только входит в состав всех современных дистрибутивов Linux, но и устанавливается по умолчанию. А во FreeBSD его можно установить из портов, если возникнет такая необходимость.

Пакет OOo разработан на базе StarOffiice, который был приобретен компанией Sun Microsystems. После этого Sun выпустила пакет OOo — бесплатный и с открытым кодом. Кстати, сейчас Sun принадлежит корпорации Oracle, поэтому не удивляйтесь, когда увидите логотип Oracle на заставках последних версий OOo.

Поскольку OOo распространяется по лицензии GPL, его вы можете использовать абсолютно бесплатно и установить на любое количество компьютеров вашей сети. Тип вашей организации может быть абсолютно любым — вы можете установить данный офисный пакет как на компьютеры государственной, так и коммерческой организации. Да хоть на свой домашний компьютер — разницы никакой нет, поскольку GPL не накладывает каких-либо ограничений на распространение программ.

В состав пакета входят следующие программы:

- OOo Writer — аналог MS Word;
- OOo Calc — аналог MS Excel;
- OOo Impress — аналог PowerPoint;
- OOo Draw — отдельный продукт, векторный графический редактор;
- OOo Base — тоже отдельный продукт, система управления базами данных (не устанавливается по умолчанию, поэтому если она вам нужна, то придется ее установить вручную из репозиториев Linux).

16.2. Оптимизация OpenOffice.org

В этой небольшой главе мы не будем рассматривать использование OOo. Согласитесь, было бы глупо в книге по администрированию UNIX рассматривать основы работы с документами и электронными таблицами. Не думаю, что человек, осиливший установку и настройку FreeBSD, не справится с использованием офисного пакета, который как две капли воды похож на MS Office, правда, несколько ранних версий. Но такой интерфейс, как по мне, самый удачный.

Задача администратора — установить офисный пакет (или другую программу), а использовать его — прерогатива пользователя. Например, проработав несколько лет на предприятии, где активно использовалась 1С, я так ее и не изучил, но мог установить, переустановить и сделать резервную копию базы данных 1С, а больше от администратора и не требуется.

OOo уже будет установлен — он устанавливается автоматически при установке Linux. Вы, как администратор, можете оказать небольшую услугу пользователям — оптимизировать OOo. Оптимизация означает ускорение запуска и настройку OOo на работу с документами формата MS Office по умолчанию. Ведь

далеко не все предприятия перешли на OOo, и чтобы пользователю каждый раз при сохранении нового документа не выполнять команду Сохранить как и не выбирать формат MS Office, вы можете установить этот формат по умолчанию[1].

Начнем с ускорения запуска. Запустите любое приложение из пакета OOo, например Приложения ▸ Офис ▸ OpenOffice.org ▸ Редактор текстов. Выполните команду Сервис ▸ Параметры, перейдите в раздел Память. Установите параметры так, как показано на рис. 16.1. Не забудьте включить режим Использовать быстрый запуск. Вот и все. Можете закрыть программу и запустить ее снова — увидите, насколько быстрее она стала запускаться. Вы даже не успеете заметить заставку!

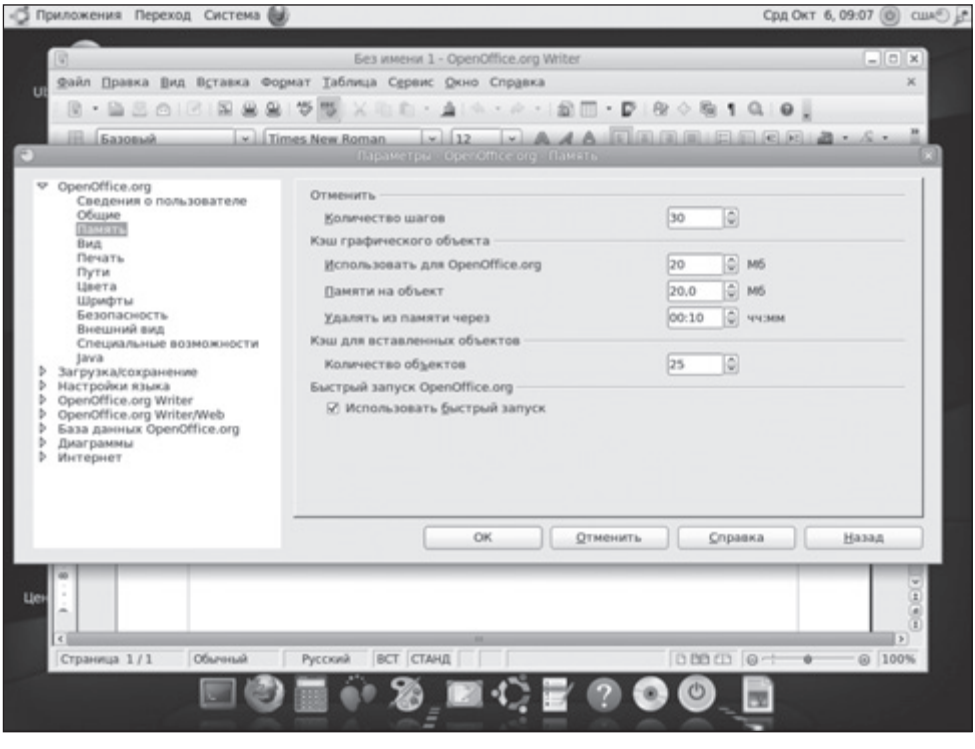

Рис. 16.1. Оптимизация запуска OOo

Затем перейдите в раздел Безопасность и нажмите кнопку Безопасн. макросов. В появившемся окне выберите низкий уровень безопасности (рис. 16.2). Не беспокойтесь, вроде бы опасных вирусов для Linux пока еще в природе не существует.

Осталось выбрать формат по умолчанию. Для этого перейдите в раздел Загрузка/Сохранение ▸ Общие. *Для каждого типа документа* (Текстовый документ, Электронная таблица, Презентация и т. д.) выберите соответствующий ему формат

[1] Но здесь лучше все же сначала выяснить у пользователей, какие именно форматы по умолчанию им нужны на самом деле. Возможно, что нужными окажутся именно форматы ODF, а значит, эту часть «оптимизации» лучше не делать.

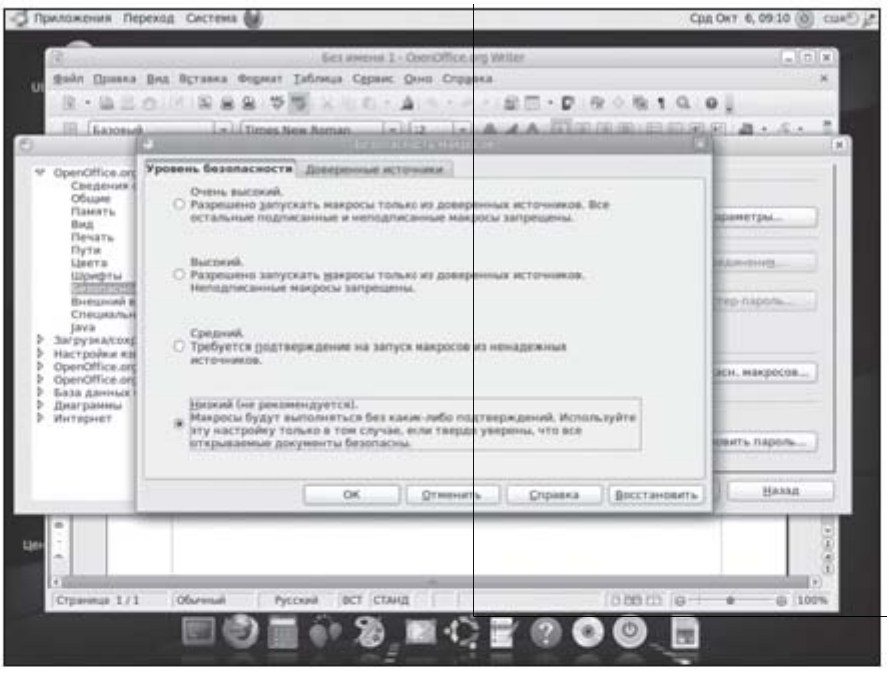

Рис. 16.2. Обеспечение нормального запуска макросов

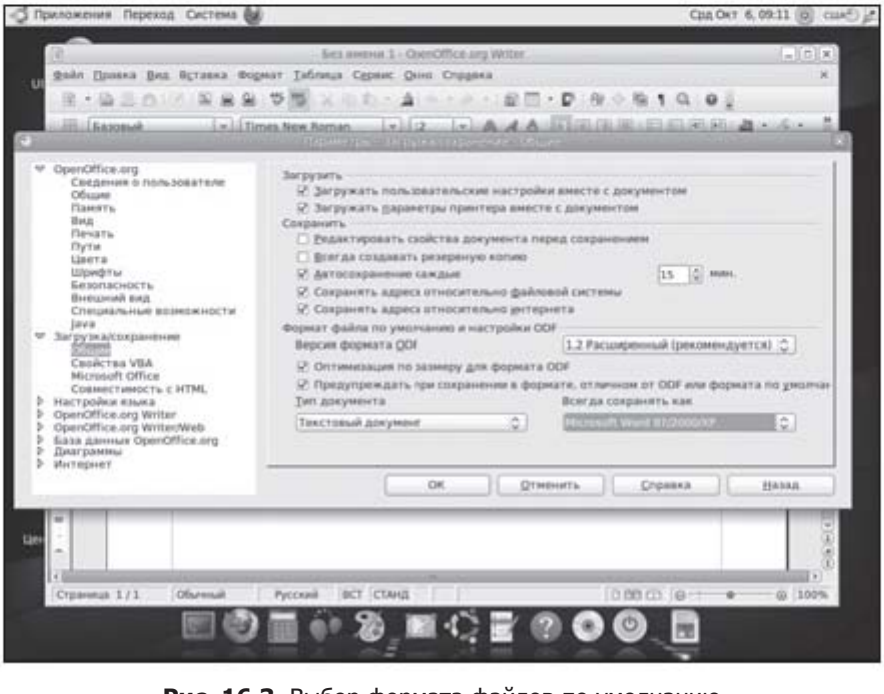

Рис. 16.3. Выбор формата файлов по умолчанию

MS Office (рис. 16.3). Поскольку я часто работаю с MS Office 2003, то я выбрал формат Microsoft Word 97/2000/XP для текстового документа. Но вы можете выбрать формат более новой версии MS Office. Все новые документы будут автоматически сохраняться в выбранном вами формате.

Вот теперь наш OOo полностью готов к работе: мы ускорили его запуск, обеспечили выполнение макросов и выбрали формат документов по умолчанию. Осталось воспроизвести все эти действия на остальных компьютерах сети...

Эмулятор wine. Запуск Windows-приложений в Linux

17

17.1. Знакомство с wine. Таблица Linux-аналогов программ

В Linux имеется возможность запуска Windows-приложений с помощью эмулятора wine. Хотя аббревиатура WINE расшифровывается как «Wine Is Not an Emulator», что означает «Wine — не эмулятор», но мы будем называть его эмулятором. Нет, конечно, можно его еще называть «средством запуска Windows-программ в Linux», но название «эмулятор» для этой программы мне кажется более удобным.

Проекту wine уже много лет — первая версия, умеющая запускать 16-битные Windows-приложения в Linux, появилась в 1993 году. Позже эмулятор научился запускать 32-битные приложения, а с 2005 года — и 64-битные. Подробно рассматривать историю wine мы не будем — не думаю, что вам интересна история, лично мне интересен результат.

Изначально wine не мог запускать игры и другие приложения, требующие для своего запуска DirectX. На базе эмулятора wine был создан эмулятор wineX, позволяющий запускать игры. Чуть позже этот эмулятор был переименован в Cedega и стал коммерческим. Но со временем поддержка DirectX появилась и в wine, причем некоторые игры выполняются в wine быстрее, чем в Cedega, поэтому сейчас нет необходимости покупать Cedega.

Сразу хочу заметить, что не каждое приложение получится запустить в wine. К тому же некоторые приложения могут работать некорректно. Тут все зависит от случая и от выбранного вами приложения. Многие игры, наоборот, запускаются без особых проблем. Но производительность игр, запущенных в wine, будет ниже. Вы же не надеялись, что игры, запущенные в wine, будут работать быстрее, чем в родной операционной системе? Конечно, играть вы все же сможете, но похвастаться высоким значением fps — вряд ли. Ради интереса в Windows Vista я установил эмулятор VMWare, в нем — гостевую операционную систему Debian, в ней — эмулятор Wine. Моя любимая «стрелялка» была запущена. Подтормаживала

исходно из-за двух эмуляторов, но играть было можно. Потом эта же игра была запущена в Linux + wine — мне показалось, что она просто летает. Так что все относительно.

Многие пользователи, узнав, что есть эмулятор wine, пытаются запустить в нем любимую программу, не зная о существовании Linux-аналога. Попробуйте сначала использовать Linux-аналог, а затем уже пытайтесь запустить Windows-программу в Linux. Программ для Linux — очень много. В таблице 17.1 я собрал лишь те, которые, на мой взгляд, заслуживают вашего внимания.

Таблица 17.1. Linux-аналоги Windows-программ

Windows-программа или класс программ	Linux-аналог
Браузер Mozilla Firefox	Mozilla Firefox для Linux — входит в состав многих дистрибутивов по умолчанию
Браузер Opera	Opera для Linux, можно скачать с www.opera.com
Почтовый клиент (MS Outlook, The Bat!, Mozilla Thunderbird)	Evolution, Mozilla Thunderbird для Linux, Sylpheed
Torrent-клиент	Transmission
Менеджер загрузки файлов (FlashGet, Download Master)	D4X (Downloader for X), FlashGot (расширение для Firefox), Gnome Transfer Manager, Kmago, GetLeft
Клиент передачи мгновенных сообщений (ICQ, QIP)	Pidgin, Empathy
FTP-клиент (FileZilla и др.)	FileZilla для Linux, gftp
IRC-клиент	Xchat, Sirc, Ksirc
Jabber-клиент	Gabber
Microsoft Word	OOo Writer
Microsoft Excel	OOo Calc
Microsoft PowerPoint	OOo Impress
Microsoft Access	OOo Base
3D Max	Blender3D, Wings 3D, Equinox-3D, Misfit Model 3D, Sharp 3D
Photoshop	GIMP
Векторный редактор	OOo Draw, Inkscape
Total Commander	mc, Gnome Commander
DC++	Linux DC++
Запись CD/DVD	Nero для Linux (коммерческая), Brasero, K3B
Архиваторы WinRAR, WinZIP	Xarchiver, PeaZIP
Виртуальная машина (VMWare)	VirtualBox
GoogleEarth	GoogleEarth для Linux
Проигрыватели мультимедиа	MPlayer, xine, Totem, Amarok, Rhytmbox, Audacious
Видео- и аудиоредакторы	Pitivi, kino, Audacity
Работа с ISO-образами	Gmount-iso
Просмотр PDF	Evince, Acrobat Reader for Linux
Создание PDF	OOo Writer
Просмотр и управление фотографиями	F-Spot, Pikasa

Windows-программа или класс программ	Linux-аналог
Сканирование изображений	Xsane
Skype	Skype
QuarkXPress	Scribus
AutoCAD; MathCAD	qCAD, BRL-CAD, gCAD3D, FreeCAD

На сайте http://www.linuxrsp.ru/win-lin-soft/table-rus.html собрана наиболее полная таблица Linux-аналогов программ, однако имейте в виду, что она не обновлялась с 2006 года, поэтому местами может быть неактуальной. В табл. 17.1 приведены наиболее актуальные для обычного пользователя программы.

17.2. Использование wine

Первым делом нужно установить wine. Для этого воспользуйтесь рекомендациями из главы 11. Затем нужно настроить wine на прозрачный запуск Windows-программ: когда вы дважды щелкаете на исполнимом (.exe) файле, программа должна запуститься — чтобы не было никакой разницы между запуском Linux-программы и Windows-программы. Для этого щелкните правой кнопкой мыши на исполнимом файле и выберите команду Открыть в другой программе... (рис. 17.1).

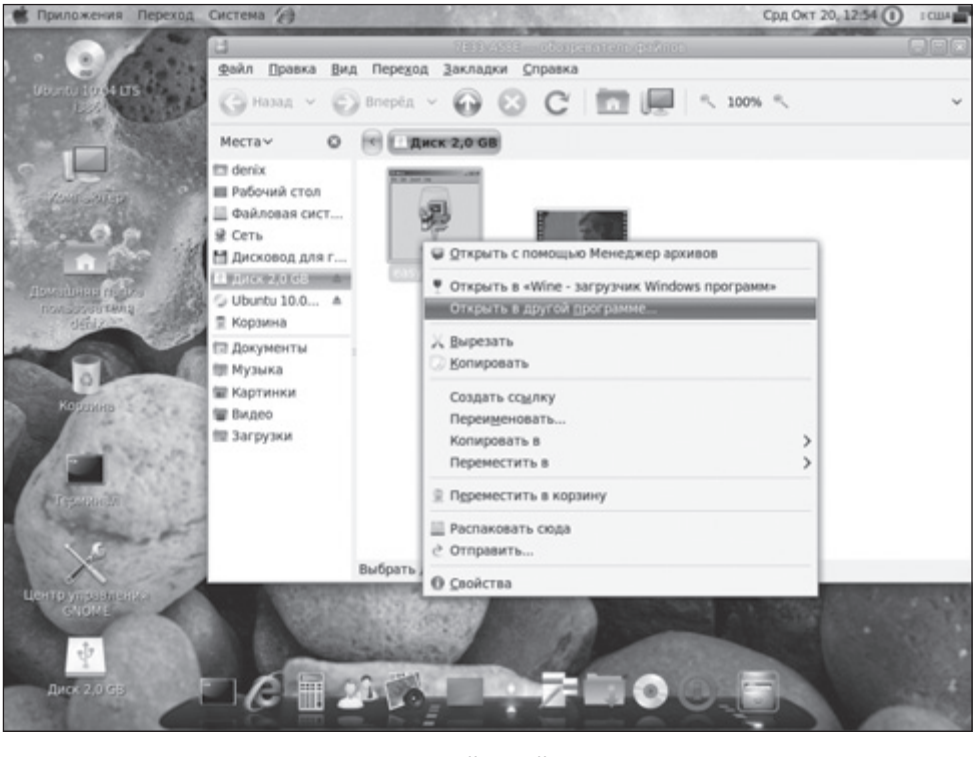

Рис. 17.1. Файловый менеджер

Можно, конечно, выбрать команду Открыть ▸ "Wine - загрузчик Windows программ", но скоро она вам надоест. В появившемся окне (рис. 17.2) нужно выбрать программу Wine - загрузчик Windows программ и нажать кнопку Открыть. После этого все exe-файлы будут запускать в вашей системе по двойному щелчку.

Все иллюстрации в этой главе соответствуют графической среде GNOME. В KDE последовательность действий будет такая же, но названия некоторых команд будут отличаться.

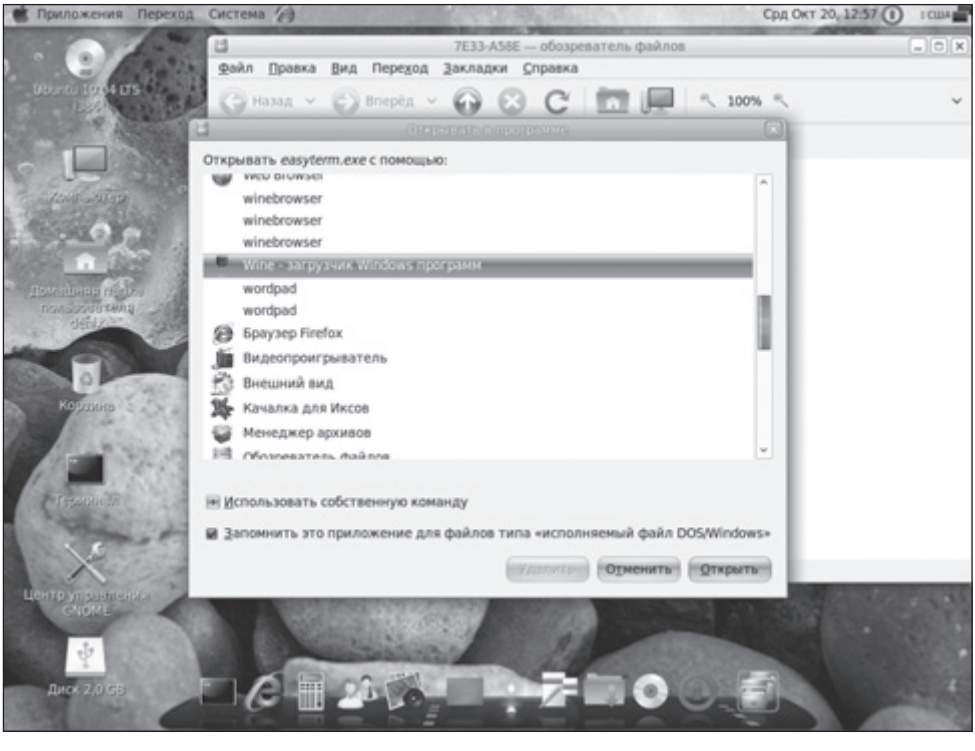

Рис. 17.2. Выбор программы для запуска exe-файлов

На рис. 17.3 запущен инсталлятор программы EasyTerm. Подтвердите установку программы (рис. 17.4). Далее установка программы ничем не отличается от аналогичного процесса, но в Windows. Просто следуйте инструкциям инсталлятора.

После установки программы команду запуска программы можно будет найти в меню Приложения ▸ Wine (Эмулятор Wine — в Denix) ▸ Программы (рис. 17.5).

Но еще раз отмечу, что далеко не все программы корректно устанавливаются в Wine и сразу начинают нормально работать. Программу EasyTerm я выбрал не случайно, а чтобы продемонстрировать «чрезвычайное происшествие», которое может произойти при запуске программы.

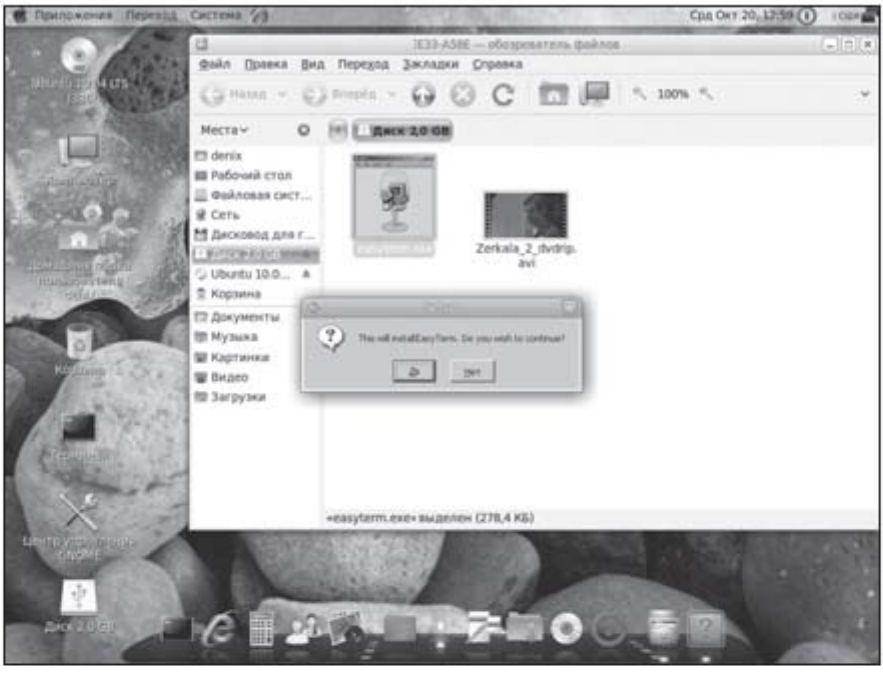

Рис. 17.3. Запущен инсталлятор программы EasyTerm

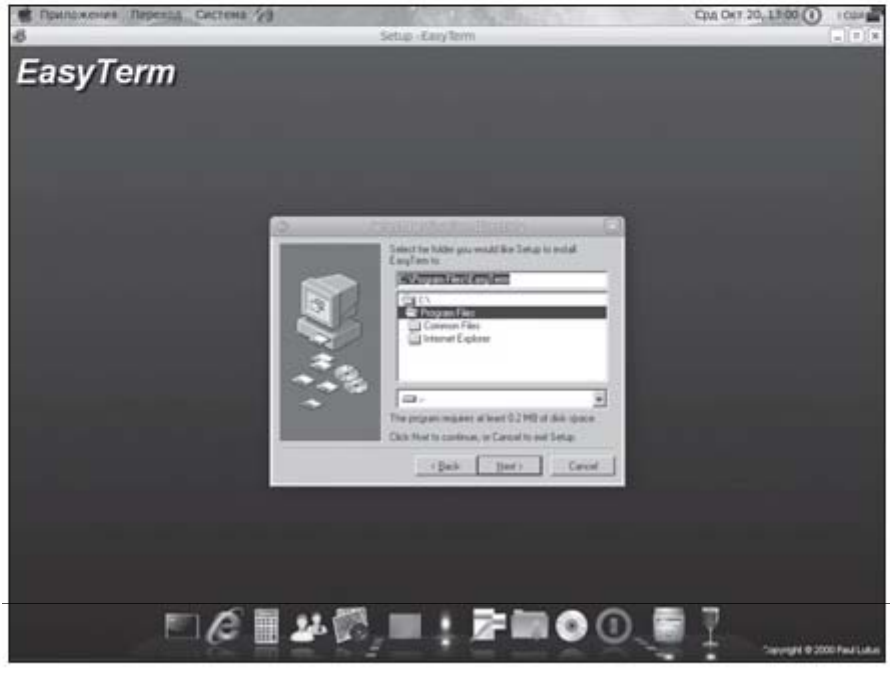

Рис. 17.4. Процесс установки EasyTerm

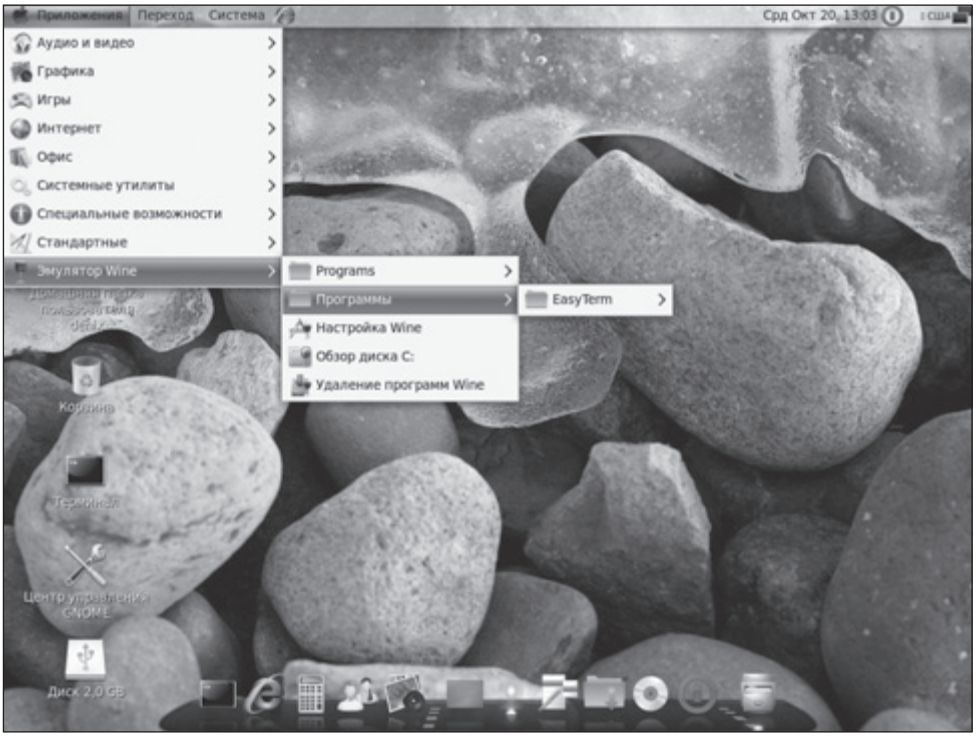

Рис. 17.5. В этом меню находятся установленные программы

Сразу после установки программы я попытался ее запустить через меню. Никакого эффекта. Еще одна попытка — тоже не увенчалась успехом. Тогда я открыл терминал, перешел в следующий каталог:

```
cd ~/.wine/drive_c/Program\ Files/EasyTerm
```

Каталог ~/.wine/drive_c для эмулятора Wine — это виртуальный диск C. Следовательно, на нем будут каталоги Windows, Program Files и некоторые другие. В каталог Program Files обычно и устанавливаются программы. Захожу в Promgram Files/EasyTerm и вижу файлы установленной мною программы. Пытаюсь запустить эту программу в консоли, чтобы увидеть ошибку:

```
wine easyterm.exe
```

В результате (рис. 17.6) вижу сообщение об ошибке:

```
err:module:import_dll Library MFC42.DLL (which is needed by L"C:\\Program Files\\EasyTerm\\
easyterm.exe") not found
…
```

Как следует из сообщения об ошибке, программе не хватает для полного счастья библиотеки MFC42.DLL. Что делать дальше, наверное, вы догадались. Правильно, «идем» на компьютер с Windows, копируем все необходимые библиотеки

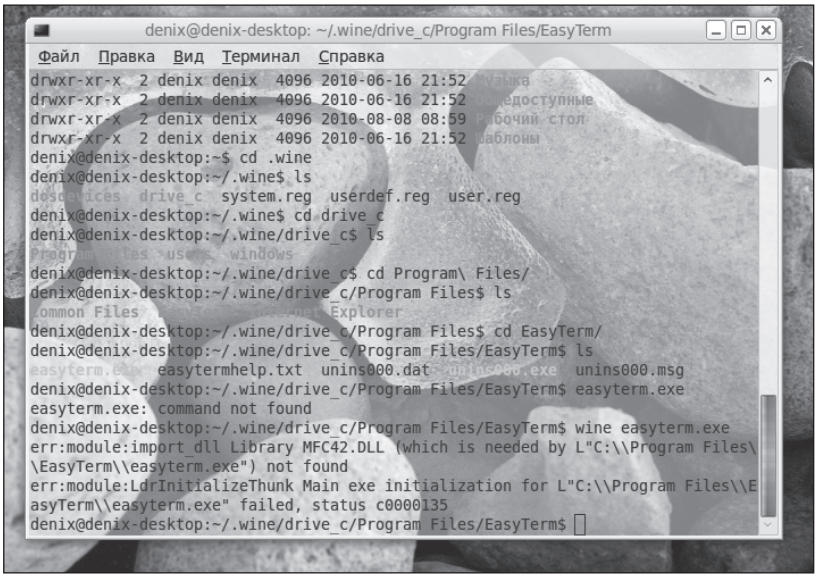

Рис. 17.6. Программа не запускается

(я также еще скопировал библиотеку MFC42RUS.DLL — на всякий случай), затем их нужно скопировать в каталог ~/.wine/drive_c/Windows/System32. После этого повторим запуск программы. На рис. 17.7 видно, что запуск прошел успешно (ни одного сообщения об ошибке), а на рис. 17.8 изображена сама программа EasyTerm.

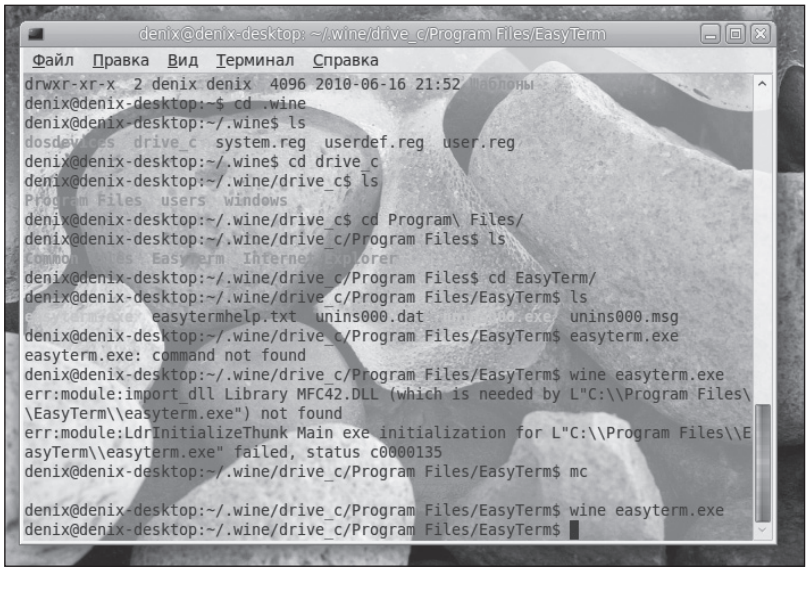

Рис. 17.7. Нормальный запуск программы

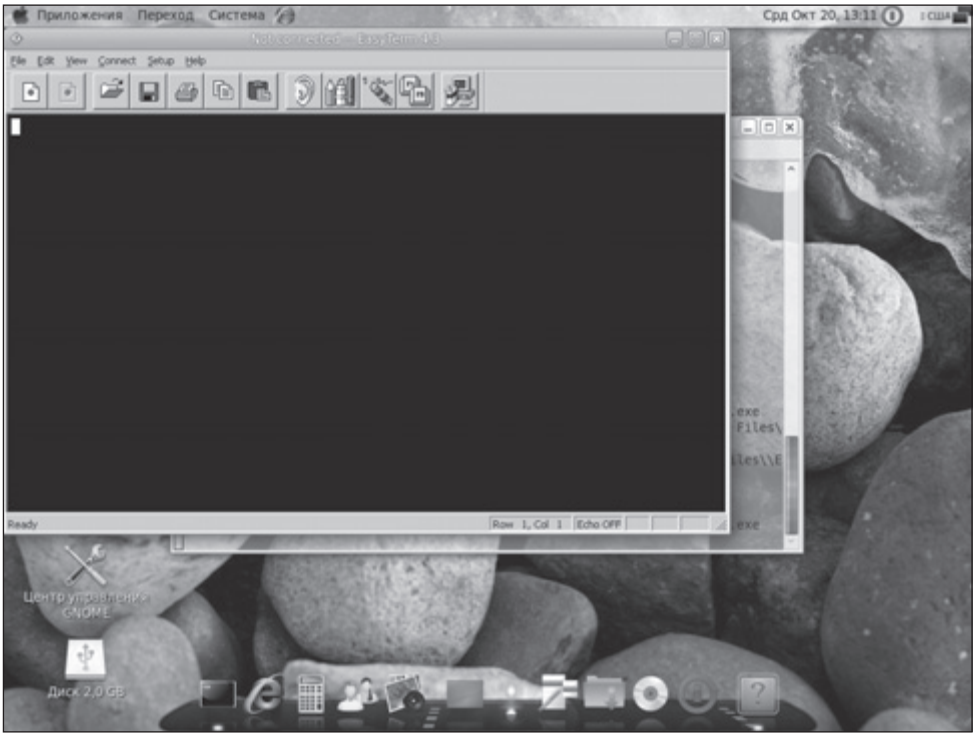

Рис. 17.8. Программа EasyTerm успешно запущена в Linux

17.3. Настройка wine и удаление программ

В программной группе wine вы найдете команду настройки wine, открывающую окно конфигурации (рис. 17.9), команду просмотра диска C: и команду удаления установленных программ. Окно настройки позволяет установить различные параметры wine, даже можно выбрать эмулируемую операционную систему.

На вкладке Графика (рис. 17.10) можно установить некоторые графические параметры, например включить или выключить шейдеры. Но настоятельно не рекомендую отключать параметр Разрешить менеджеру окон управлять окнами wine. Тогда закрыть программу можно будет только средствами самой программы, то есть нажать кнопку Выход или подобную, но если программа зависнет, вам придется завершать процесс `wine` командой `kill`.

Если в игрушке не будет слышен звук, перейдите на вкладку Аудио и выберите ALSA драйвер (рис. 17.11). Также нажмите кнопку Проверить звук. Проблемы со звуком бывают, если выбран драйвер, отличный от используемого вашей системой. По умолчанию Linux использует ALSA.

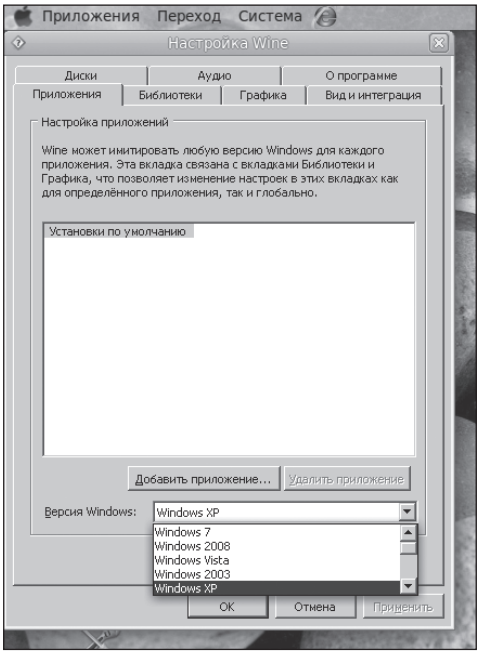

Рис. 17.9. Windows 7 тоже поддерживается wine

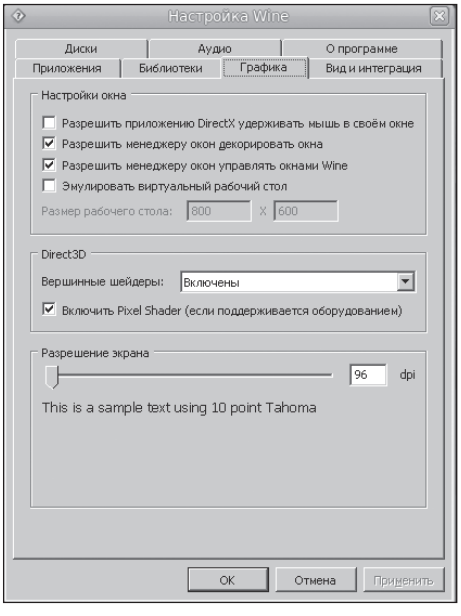

Рис. 17.10. Вкладка Графика

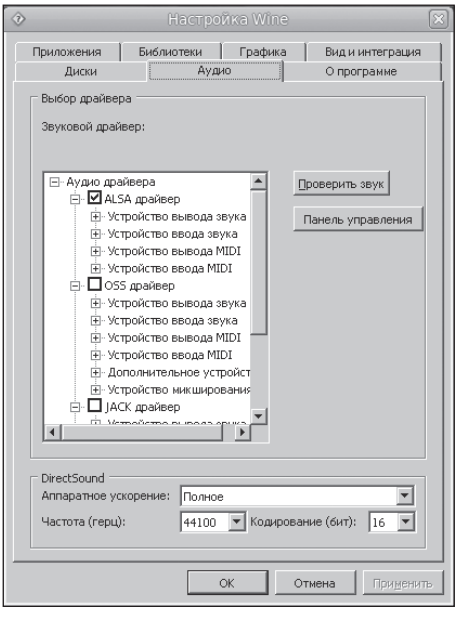

Рис. 17.11. Параметры звука

Команда обзора диска C: открывает файловый менеджер (рис. 17.12), отображающий содержимое папки ~/.wine/drive_c. С тем же успехом можно было открыть файловый менеджер из меню Переход — как кому больше нравится.

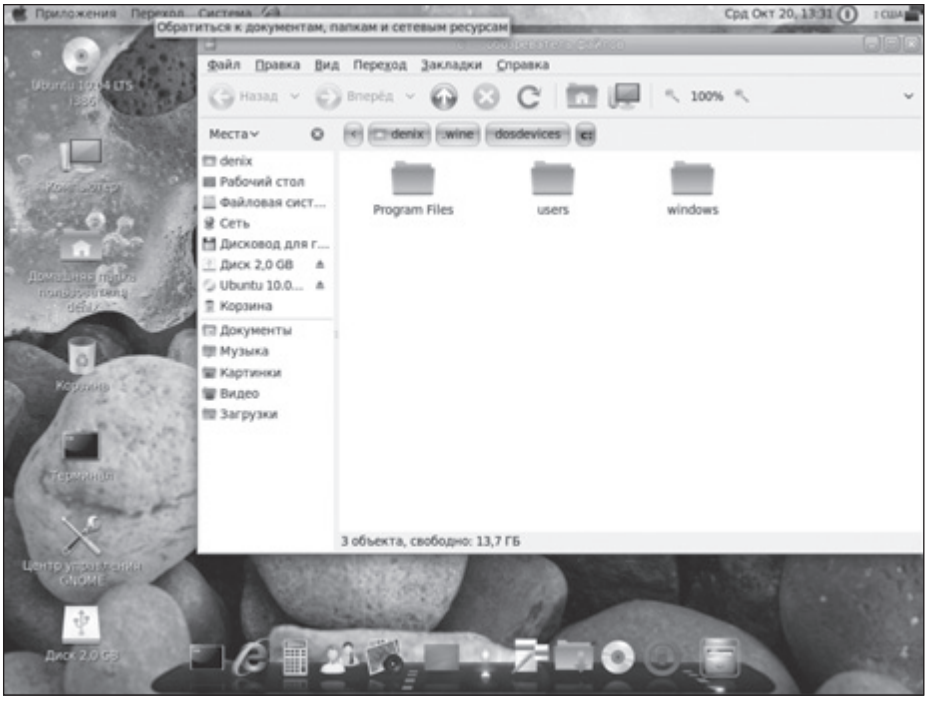

Рис. 17.12. Обзор виртуального диска C:

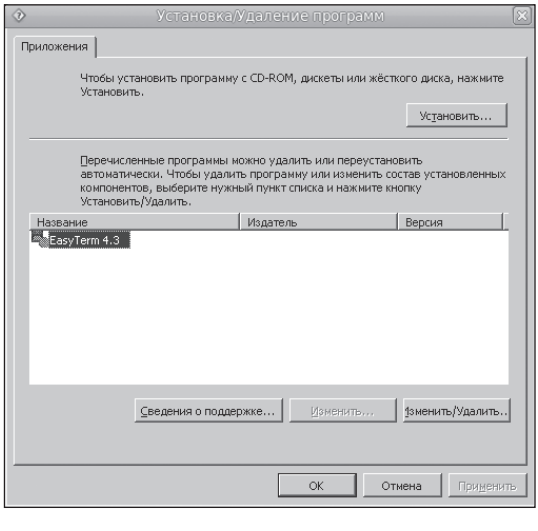

Рис. 17.13. Окно удаления программ

Удалять установленные программы нужно только с использованием команды Удаление программ wine (в других дистрибутивах она может называться иначе, но суть будет та же) (рис. 17.13). Окно удаления программ напоминает аналогичное окно в Windows, так что, думаю, с ним вы разберетесь сами.

ПРИМЕЧАНИЕ

Все снимки экрана были сделаны в моем дистрибутиве Denix 3.0.2 (http://denix.dkws.org.ua), основанном на базе Ubuntu.

Запуск 1С в Linux

18

18.1. О чем эта глава

Данная глава посвящена, как обычно, экономии. Версия 1С:Предприятие для Linux вызывает повышенный интерес у администраторов, поскольку позволяет сэкономить для предприятия стоимость Windows Server и Microsoft SQL Server. Сколько стоит Windows Server, мы уже знаем, а вот стандартная редакция SQL Server стоит почти 1000 долларов, стоимость Enterprise-версии стоит почти 28 тысяч долларов, поэтому есть над чем задуматься. Для малого бизнеса предназначена самая дешевая версия (MS SQL Server for Small Business) стоимостью 780 долларов, но для малого бизнеса — это тоже немалая сумма.

Сэкономить можно и на выборе версии 1С:Предприятия: 64-битная версия стоит почти в два раза дороже, чем 32-битная (а именно на 30 000 рублей), поэтому рекомендую из соображений экономии выбрать 32-битную версию, тем более что она должна работать стабильнее 64-битной.

В этой главе мы поговорим об установке серверной Linux-версии 1С, а не о запуске «пиратской» версии 1С в Linux, как можно было подумать из названия главы. В случае с 1С установка не ограничивается только лишь инсталляцией пакетов. Нужно произвести определенную подготовительную работу, о которой и пойдет речь в этой главе.

18.2. Подготовка к установке 1С

Первым делом нужно получить саму 1С. Можно заказать диски, но намного проще купить ее на сайте http://users.v8.1c.ru/ — вы сразу скачаете электронную версию, что позволит перейти к установке, а не ждать, пока вы получите диски. Вам нужно взять версии пакетов, совместимые с вашим дистрибутивом: RPM для Fedora, Mandriva, SUSE, DEB — для Debian и Ubuntu.

Лучше поручить работу с 1С вашему помощнику или любому другому человеку; пока он будет заниматься оплатой и прочими нюансами, вы установите отдельный жесткий диск и подмонтируете его к каталогу /var — на этом жестком

диске и будет храниться БД предприятия. Если есть возможность, лучше организуйте RAID-массив (RAID5) для обеспечения отказоустойчивости данных.

Для работы 1С нужен драйвер HASP-ключа. Без него 1С не сможет обслуживать более 12 клиентов. Если у вас меньше клиентов, то можете обойтись и без драйвера (и без ключа). Драйвер можно скачать по адресу:

ftp://ftp.aladdin.com/pub/hasp/srm/Linux/HASP_SRM_LINUX_3.50_Run-time_Installer_script.tar.gz

Кроме 1С и драйвера HASP вам понадобится SQL-сервер. Мы будем использовать PostgreSQL. Но версия из репозитория дистрибутива нам не подходит, поскольку версия из репозитория не блокирует таблицы в автоматическом режиме, как это делает MS SQL и что необходимо для 1С. Поэтому нам нужна пропатченная версия для 1С, которую можно получить по адресу:

ftp://updates.etersoft.ru/pub/Etersoft/Postgres@Etersoft/stable/

По этому адресу вы найдете нужную версию PostgreSQL для самых разных дистрибутивов Linux. Установите PostgreSQL, после чего отредактируйте /etc/sysctl.conf — в него нужно добавить две строки:

```
kernel.shmall=134217728
kernel.shmmax=134217728
```

Сохраните /etc/sysctl.conf и введите команду:

```
sudo sysctl -p
```

Почти все готово для запуска PostgreSQL, нужно только установить одну библиотеку:

```
sudo apt-get install libxslt1.1
```

Запустим SQL-сервер:

```
sudo /etc/init.d/postgresql start
```

Наша задача — изменить пароль главного пользователя. Откройте файл /var/lib/pgsql/data/pg_hba.conf и найдите в нем строку:

```
local all all ident sameuser
```

Ее нужно изменить так:

```
local all all trust
```

Перезапустите PostgreSQL:

```
sudo /etc/init.d/postgresql restart
```

Теперь изменим пароль главного пользователя:

```
psql -U postgres -d template1 -c "ALTER USER postgres PASSWORD 'пароль'"
```

После этого опять нужно отредактировать /var/lib/pgsql/data/pg_hba.conf и вернуть все как было, то есть строку

```
local all all trust
```

заменить строкой

```
local all all ident sameuser
```

18.3. Установка 1С и драйвера HASP

После установки пакетов с 1С нужно установить права к каталогу /opt/1c:

```
sudo chown -R usr1cv81:grp1cv81 /opt/1C
```

Затем нужно обеспечить автоматический запуск сервера 1С:

```
sudo update-rc.d srv1cv81 defaults
```

Чтобы при работы с БД 1С у вас не было ошибок, нужно правильно установить локаль системы. Привожу команды для дистрибутива Debian и совместимых с ним:

```
locale-gen en_US
locale-gen ru_RU
dpkg-reconfigure locales
```

Для установки драйвера ключа HASP нужно выполнить команды:

```
tar xzvf HASP_SRM_LINUX_3.50_Run-time_Installer_script.tar.gz
./dinst .
sudo echo 'none /proc/bus/usb usbfs defaults 0 0' >> /etc/fstab
sudo reboot
```

Практически все уже настроено. Осталось перенести БД с существующего 1С-сервера под управлением Windows на наш только что созданный Linux-сервер. Откройте оснастку Серверы 1С Предприятия, выберите Центральные серверы 1С Предприятия 8.1. Затем выполните команду меню Действие ▸ Создать. Создайте новый сервер, создайте новую базу и перенесите в нее данные из 1С. По большому счету, далее следует обратиться к руководству 1С, поскольку вся дальнейшая настройка относится не к настройке UNIX-сервера, а к настройке непосредственно 1С.

IV

Миграция сервера на UNIX

Прежде чем приступить к настройке сервера, важно понимать разницу между сервером (физическим компьютером, предоставляющим какие-то сервисы другим компьютерам сети) и программой-сервером, запущенной на компьютере-сервере. Так, у вас может быть один физический сервер, но он может одновременно являться и веб-сервером, и FTP-сервером и почтовиком. А все это потому, что на этом компьютере установлено три программы-сервера: Apache, ftpd и sendmail (программы могут быть другими — все зависит от того, с какими программами предпочитает работать администратор). Сервер — это просто «железка», без «математики», то есть программного обеспечения, он полностью бесполезен. А когда на этот компьютер устанавливается программное обеспечение, он может предоставлять определенные услуги другим компьютерам сети.

Теоретически, любой компьютер (даже ваш личный ноутбук) можно превратить в сервер. Но как будет работать такой сервер — это уже другой вопрос. Как правило, для сервера сети выбирают самый мощный компьютер. Хотя это вопрос относительный. Если сервер сети выполняет функции только шлюза, то есть предоставляет другим компьютерам доступ к Интернету, то вполне хватит старенького «Пентиума» с 128 Мбайт оперативной памяти, который уже несколько лет пылится в углу (и использовать нельзя, и выбросить жалко). А вот если вам нужен сервер баз данных или сервер терминалов, то вам понадобится гораздо более мощный компьютер.

Все это было сказано, чтобы вы понимали, что когда мы будем настраивать, скажем, DHCP-сервер, мы не будем настраивать его с нуля, мы будем настраивать программу, которая выполняет функции DHCP-сервера. Если вы уже состоявшийся Windows-администратор, то вы все это понимаете, а вот если вы до этого ни разу не настраивали какой-либо сервер, вы должны это «переварить».

Подключение рабочей станции под управлением UNIX к Windows-сети

19

19.1. Введение в пакет Samba

Представьте, что вы установили UNIX на свой компьютер в офисе или в домашней сети. Но большинство компьютеров в этой сети работают под управлением Windows, а это означает, что ресурсы Windows-сети (общие файлы и принтеры) будут вам недоступны. Хотя это до того момента, как вы установите пакет Samba. После его установки вы не только сможете использовать общие ресурсы Windows-сети, но еще и предоставлять их. Да, вы сможете дать удаленным пользователям возможность использовать файлы и принтеры вашего компьютера (понятно, вы можете задать ограничения доступа). Остальные пользователи даже не заметят, что вы работаете под UNIX. Для них ваш компьютер будет обычной рабочей станцией, как и все остальные узлы сети.

Кроме того, пакет Samba позволяет даже организовать сервер Windows-сети — первичный контроллер домена (PDC, Primary Domain Controller). Забавно получается: UNIX может управлять Windows-сетью.

В этой главе будет рассмотрена базовая настройка Samba в Linux и FreeBSD, а в следующих двух главах мы поговорим о настройке первичного контроллера домена и миграции с ActiveDirectory на LDAP.

Сразу хочу отметить, что настройка Samba будет производиться путем редактирования конфигурационных файлов, без использования конфигураторов, которые могут отличаться в зависимости от используемого дистрибутива и его версии.

19.2. Установка Samba

В Linux с помощью вашего менеджера пакетов нужно установить пакет samba-server. Кроме установки пакета никаких других действий предпринимать не нужно — при установке пакета сервис Samba будет автоматически настроен на запуск при загрузке системы. Управлять сервисом можно с помощью команд:

```
# service smb start
# service smb stop
# service smb restart
```

Во FreeBSD все немного сложнее. Первым делом нужно установить порт samba34:

```
# cd /usr/ports/net/samba34
# make install
```

На момент написания этих строк версия 3.4 была самой последней. Сэкономить время путем установки пакета (с помощью pkg_add) не получится, так как пакет samba34 не содержит поддержки ActiveDirectory. А уж если мы будем пытаться интегрировать наш Samba-сервер в корпоративную среду, то поддержка ActiveDirectory нам понадобится. А ее можно включить только при установке samba34 из портов (рис. 19.1).

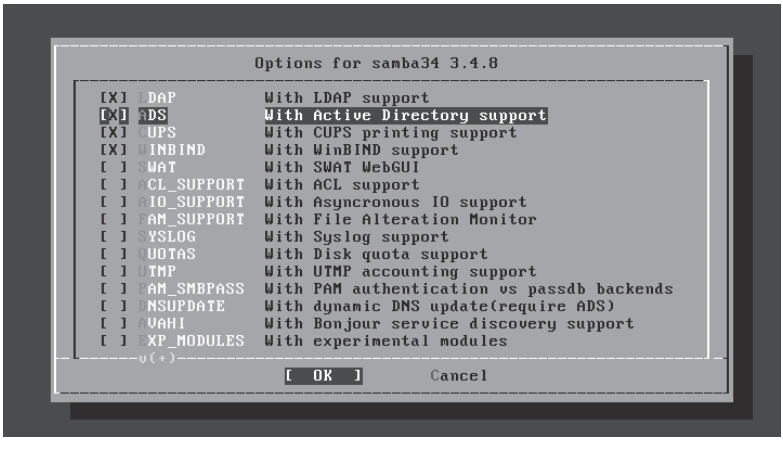

Рис. 19.1. Включение поддержки ADS при сборке Samba

После сборки или установки из пакета Samba добавьте в ваш /etc/rc.conf строку:

```
samba_enable="YES"
```

Для управления Samba используются команды:

```
# /usr/local/etc/rc.d/samba start
# /usr/local/etc/rc.d/samba stop
# /usr/local/etc/rc.d/samba restart
```

Основной файл конфигурации Samba называется smb.conf. В Linux он находится в каталоге /etc/samba, а во FreeBSD — в каталоге /usr/local/etc/.

19.3. Конфигурационный файл smb.conf

Откройте ваш smb.conf. Первым делом установим общие опции, влияющие на работу всего сервера:

```
WORKGROUP = WRKGRP
netbios name = Server
server string = UNIX computer
security = share
guest account = guest
unix charset = UTF-8
dos charset = UTF-8
display charset = UTF-8
interfaces = 192.168.1.1
```

Параметр WORKGROUP задает имя рабочей группы или домена. Параметр netbios name задает имя нашего сервера. Третий параметр (server string) — это просто комментарий, который увидят Windows-пользователи при просмотре сети. Параметр security определяет тип безопасности. Для сети без выделенного сервера больше всего подходит тип share или тип user (но тогда нужно создать Samba-пользователей и задать для них пароли командой smbpasswd). Для сетей с выделенным сервером нужно использовать тип server.

Далее указывается гостевая учетная запись. Данная учетная запись должна существовать (обычно уже существует на момент установки Samba, но лучше проверить это). После задается кодировка UTF-8 — прошли те времена, когда Samba не поддерживала UTF-8, теперь все нормально, и вместо «каракуль» вы увидите русские буквы.

Последний параметр задает адреса интерфейсов, на которых должна работать Samba, но эту опцию можно не указывать.

После изменения глобальных параметров можно приступить к определению общих ресурсов. Рассмотрим классический пример определения общего каталога:

```
[public]
# комментарий для общего ресурса
comment = Public
# каталог, предоставляемый в общее использование
path = /var/samba
# "только чтение" не используем
read only = no
# разрешаем запись
writable = yes
# разрешаем доступ гостю
guest ok = yes
# разрешаем просмотр общего каталога
browseable = yes
```

Каталог /var/samba мы предоставляем в полный доступ (чтение и запись) всем желающим. Из соображений безопасности — не самый хороший вариант, но возможно, кому-то пригодится и такой вариант, например для небольшой домашней сети, где все свои. В реальных условиях запись лучше запретить, иначе место на жестком диске может быстро закончиться:

```
[public]
# комментарий для общего ресурса
comment = Public Read-only
# каталог, предоставляемый в общее использование
path = /var/samba
# запись запрещена, разрешено только чтение файлов
```

```
read only = yes
writable = no
# разрешаем доступ гостю
guest ok = yes
# разрешаем просмотр оглавления общего каталога
browseable = yes
```

Следующий пример подходит для «экспортирования» домашнего каталога пользователей

```
[homes]
 comment = Home Directories
# запрещен просмотр оглавления ресурса [homes]
 browseable = no
# список пользователей, которым разрешен доступ
# Для секции [homes] используется значение %S для подстановки
# имени пользователя
# Если параметр valid users не определен, то подключиться могут все пользователи
 valid users = %S

# разрешаем только просмотр каталогов, запись запрещена!
 writable = no

# маска при создании файлов, нужна, если вы разрешаете запись
create mask = 0600

# маска при создании каталогов, нужна, если вы разрешаете запись
directory mask = 0700
```

Директива valid users разрешает доступ только реальным пользователям, гостевой доступ запрещен.

Теперь разберемся с общими принтерами. Чтобы иметь возможность предоставить в общее использование принтеры, нужно установить систему печати CUPS и порт cups-samba:

```
# cd /usr/ports/print/cups-samba
# make install clean
```

Используя cups-samba, очень просто управлять общими принтерами — достаточно добавить в smb.conf несколько строк, и тогда все принтеры, помеченные в printers.conf (или с помощью интерфейса управления CUPS) как общие, будут экспортированы демоном Samba.

После установки cups-samba в файле smb.conf нужно найти и раскомментировать (они обычно закомментированы) следующие строки:

```
load printers = yes
printing = cups
printcap name = cups

[printers]
comment = All Printers
path = /var/spool/samba
browseable = no
public = yes
guest ok = yes
writable = no
printable = yes
```

```
printer admin = root

[print$]
comment = Printer Drivers
path = /usr/local/share/cups/drivers
browseable = yes
guest ok = yes
read only = yes
write list = root
```

Первые три строки указывают, что нужно загрузить принтеры из CUPS, а затем следует определение двух общих ресурсов — все принтеры и ресурс с драйверами принтеров. После этого нужно перезапустить Samba и добавить принтеры:

```
# cupsaddsmb -U root имя_принтера
```

Имя принтера должно быть определено в printers.conf (см. главу 13), например:

```
# cupsaddsmb -U root samsung
```

19.4. Клиент Samba–smbclient

Программа smbclient позволяет использовать общие ресурсы Windows-компьютеров. В Linux этой программой вам пользоваться не придется, поскольку есть более удобные средства просмотра общих ресурсов, уже интегрированные в файловые менеджеры графических систем GNOME и KDE. А вот на BSD-сервере, возможно, вам захочется обратиться к ресурсам Windows-сети. Для этого можно использовать программу smbclient, формат вызова которой в самом простом случае выглядит так:

```
$ smbclient \\сервер\ресурс [пароль]
$ smbclient \\сервер\ресурс [пароль] -U пользователь
```

Программа smbclient похожа на программу ftp — она использует те же команды, что и программа ftp. Например, вы можете использовать команды get и mget для копирования файлов с Windows-ресурса и команды put и mput для записи файлов на Windows-ресурсы при наличии соответствующих прав.

Первичный контроллер домена 20

20.1. Постановка задачи

В этой главе мы настроим Samba на базе FreeBSD как первичный контроллер домена (PDC), чем, по сути, заменим Windows Server и сэкономим немного денег для предприятия. А может даже и много. Чего греха таить? Штраф за использование нелицензионного программного обеспечения может превышать стоимость самого программного обеспечения, поэтому если пока никто не узнал, что у вас установлена нелицензионная версия Server, значит, у вас есть два варианта. Первый — это быстренько побежать к ближайшему дилеру Microsoft и приобрести лицензионную версию. Вот только не забудьте о количестве лицензий. Недостаточно купить лицензионную версию Windows Server, нужно еще заплатить за количество пользователей, которые могут использовать ваш сервер. Например, сама Windows 2008 Server Standard стоит 737 долларов (во всяком случае, стоила на момент написания этих строк у одного из дилеров), а эта же операционная система с лицензией на 10 клиентов — в два раза дороже.

Второй способ заключается в создании PDC на базе уже установленной (я надеюсь) на одном из компьютеров вашей сети FreeBSD. Зачем нам нужен PDC в UNIX-сети? Давайте будем смотреть правде в глаза. Всю сеть перевести на UNIX одним махом не получится — это требует времени, да и некоторые компьютеры из-за установленного программного обеспечения не получится перевести на UNIX вообще. Например, я еще не видел нормальной системы автоматического проектирования для UNIX. А она используется на многих предприятиях. Получается, что компьютеры конструкторов в ближайшее время останутся под управлением Windows. С рабочими станциями проще: многие из них покупались с предустановленными OEM-версиями Windows, поэтому даже в случае проверки с лицензиями все будет в порядке. А вот на программном обеспечении для сервера на просторах бывшего СССР принято экономить. Как уже было отмечено, за версию Windows 2008 Server на 10 клиентов нужно отдать почти 1500 долларов. А сам «сервер» (компьютер, на который должна быть установлено ПО за 1500 долларов) стоит во многих случаях дешевле. Ведь по сути, что такое сервер? Это компьютер с более мощным (не всегда) процессором, большим количеством

оперативной памяти, материнской платой с поддержкой RAID и несколькими жесткими дисками на пару терабайтов. Да, двухъядерник от Intel (процессоры от AMD больше греются, поэтому не думаю, что их будут использовать на серверах разумные администраторы), 3–4 Гбайт оперативной памяти и дисковый массив из двух винчестеров по 500–1024 Гбайт каждый. Вот это и есть конфигурация обычного сервера.

Наша задача ясна — настроить первичный контроллер домена на базе FreeBSD с использованием пакета Samba, с которым вы уже успели познакомиться в прошлой главе.

20.2. Установка Samba

Пакет Samba нужно установить из портов:

```
# cd /usr/ports/net/samba3
# make install clean
```

Для первичного контроллера домена нам нужно включить следующие опции при настройке Samba:

```
[X] WINBIND      With WinBIND support
[X] ACL_SUPPORT  With ACL support
[X] SYSLOG       With Syslog support
[X] QUOTAS       With Disk quota support
[X] UTMP         With UTMP accounting support
[X] POPT         With system-wide POPT library
[X] PCH          With precompiled headers optimization
```

Лучше выбрать версию Samba 3.0, поскольку в более поздних есть проблемы с выполнением сценариев (см. далее). Решение этих проблем не стоит потраченного на это время — ведь можно просто использовать версию 3.0.

20.3. Редактирование файла smb.conf

Далее нужно отредактировать файл конфигурации /usr/local/etc/smb.conf. Его полностью рабочий листинг с моими комментариями представлен далее (листинг 20.1).

Листинг 20.1. Файл smb.conf

```
 [global]
   workgroup = LTD
   netbios name = PDC
   server string = Primary Domain Controller

# Определяем сценарии (действия), характерные для сервера

# Действие при добавлении пользователя
# Следующие две строки на самом деле являются одной строкой
   add user script = /usr/sbin/pw useradd "%u" -d /dev/null
  -s /sbin/nologin -L "russian" -m -g ntusers -c "%u"
```

продолжение ⇩

Листинг 20.1 *(продолжение)*

```
# Сценарий для создания учетной записи компьютера
# Следующие две строки являются одной строкой!
   add machine script = /usr/sbin/pw useradd "%u" -d /dev/null
        -s /sbin/nologin -L "russian" -m -g computers -c "computer_account"

# Сценарий для удаления пользователя
   delete user script = /usr/sbin/pw userdel "%u" -r

# Сценарий для переименования пользователя
   rename user script = /usr/sbin/pw usermod "%uold" -l "%unew"

# Сценарий для перезапуска Samba
   shutdown script = /usr/local/etc/rc.d/samba restart &

# Сценарий создания новой группы
   add group script = /usr/sbin/pw groupadd "%g"

# Сценарий удаления группы
   delete group script = /usr/sbin/pw groupdel "%g"

# Сценарий добавления пользователя в группу
   add user to group script =/usr/sbin/pw groupmod "%g" -m "%u"

# Сценарий установки группы по умолчанию для пользователя
   set primary group script = /usr/sbin/pw usermod "%u" -g "%g"

# Сценарий для удаления пользователя из группы
   delete user from group script =/usr/sbin/pw groupmod "%g" -d "%u"


# SAM-база и кодировки
   passdb backend = tdbsam:/usr/local/etc/samba/passdb.tdb
   display charset         = koi8-r
   unix charset            = koi8-r
   dos charset             = koi8-r


# Сценарии, выполняемые при входе в домен
   logon script = scripts\logon.bat
# Разрешить вход в домен
   domain logons = Yes

# Обратите внимание на следующие настройки: они влияют на то,
# будет ли наш сервер первичным контроллером домена

# Чтобы "победить" Windows-серверы, то есть чтобы наш сервер
# стал Master Browser для домена, нужно установить значение
# больше 65
   os level = 95
# Наш сервер является предпочитаемым мастер-сервером:
       preferred master = Yes
# Наш сервер является мастером для домена
   domain master = Yes
   idmap uid = 5000-9999
```

```
    idmap gid = 5000-9999

# Раскомментируйте interfaces, если нужно
    #interfaces = 192.168.1.2/24
    security = user
# Поддержка WINS
    wins support = yes
    dns proxy = yes

# Где находятся перемещаемые профили
    #logon path = \\server\profiles\%U
    logon path =
    logon home =
    template homedir =

# Параметры протоколирования
    log level = 5
    log file = /var/log/samba/log.%m

# Кто будет администратором
    admin users     = "@LTD\Domain Admins"

# Стандартные "шары"
[IPC$]
        path = /tmp


[print$]
        comment = Printer Drivers Share
        path = /usr/home/samba/drivers

[netlogon]
        path = /nethome/samba/netlogon
        read only = no
        browseable = yes

[profiles]
        path = /nethome/samba/profiles
        browseable = yes
        create mask = 0600
        directory mask = 0700
        read only = no
        guest ok = yes
```

20.4. Создание сценариев для нашего сервера

После редактирования конфигурационного файла Samba нужно создать сценарии, необходимые для полноценной работы первичного контроллера домена. Практически все действия можно заменить простыми командами, что и было сделано в листинге 20.1. Но иногда нужно усложнить сценарии. Например, вы желаете

знать, когда был добавлен тот или иной пользователь. Для этого измените add user script так:

```
add user script = /usr/local/etc/samba/add_user.sh "%u"
```

Сценарий add_user.sh будет выглядеть так:

```
#!/bin/sh

# Сценарий добавления пользователей
/usr/sbin/pw useradd "$1" -d /dev/null -s /sbin/nologin -L "russian" -m -g ntusers -c "$1"

# Ведем протокол
cd /tmp
echo "User '$@' added at `date +%Y-%m-%d` `date +%H:%M:%S`" >> addusr.log
```

В итоге информация о добавленных пользователях будет помещена в протокол /tmp/addusr.log. Не забудьте сделать сценарий add_user.sh исполнимым:

```
# chmod +x add_user.sh
```

Еще раз отмечу, что создавать сценарии необязательно — можно обойтись и одной командой, но для отладки (особенно, когда настраиваете PDC в первый раз) можно и создать.

20.5. Почти все

Запустим или перезапустим Samba:

```
/usr/local/etc/rc.d/samba start (или restart)
```

Добавим пользователя root для Samba. Вообще-то это не обязательно, но бывшим UNIX-пользователям будет привычнее:

```
smbpasswd -a root
```

Далее нужно ввести следующие команды:

```
pw groupadd ntadmins
pw groupadd ntusers
pw groupadd computers

net groupmap add ntgroup="Domain Admins" unixgroup=ntadmins rid=512 type=d
net groupmap add ntgroup="Domain Users"  unixgroup=ntusers  rid=513 type=d
net groupmap add ntgroup="Domain Guests" unixgroup=nobody rid=514 type=d
net groupmap add ntgroup="Domain Computers" unixgroup=computers type=d
```

Эти команды создают нужные группы и делают отображение (маппинг) Windows-групп на системные группы. Данные команды придется ввести раз в жизни. Больше о них вы не вспомните.

Добавим администраторов домена:

```
pw groupmod ntadmins -m root
smbpasswd -a den
pw groupmod ntadmins -m den
```

Администраторы домена — это пользователи, добавленные в группу ntadmins. Осталось ввести наш сервер в домен:

```
net join <имя компьютера>
```

Вот и вся настройка. Можете еще раз перезагрузить Samba и попробовать войти с Windows-компьютеров в домен. Пока можно будет войти под именем root. Остальных пользователей можно добавить так:

```
smbpasswd -a <имя пользователя>
```

Миграция с ActiveDirectory на LDAP

21

21.1. Прежде чем приступить к настройке

В прошлой главе мы рассмотрели, как настроить первичный контроллер домена на базе FreeBSD и Samba. В этой главе мы усложним задачу и рассмотрим миграцию с ActiveDirectory на LDAP (Lightweight Directory Access Protocol — легковесный протокол доступа к каталогам). Чтобы фанатам Linux не было обидно, в этой главе мы будем рассматривать настройку на базе Linux. Во FreeBSD форматы файлов будут теми же, но следите за именами файлов — они могут быть несколько иными (имеются в виду полные имена файлов — названия конфигурационных каталогов во FreeBSD и Linux отличаются).

Итак, предположим, что у нас есть Windows 2000 Server с настроенным ActiveDirectory, наш домен называется LTD, а NetBIOS-имя Windows-сервера — server (имя для Linux-сервера придумаем в процессе настройки). Для реализации нашего проекта нужно установить следующие пакеты: samba (желательно использовать версию 3, если получится), slapd, smbldap-tools, libnss-ldap, apache2, nscd, phpldapadmin, libpam-ldap.

В этой главе мы выполним не только настройку сервера на базе Linux, но и произведем миграцию пользователей и компьютеров с ActiveDirectory на LDAP.

В большинстве случаев у вас будет установлена русская версия Windows Server, поэтому перед началом миграции вам нужно переименовать следующие группы пользователей (в скобках приводится новое название группы):

- Администраторы домена (Domain Admins).
- Компьютеры домена (Domain Computers).
- Гости домена (Domain Guests).
- Пользователи домена (Domain Users).

В прошлой главе мы не переименовывали группы, поскольку предполагалось, что самого процесса миграции не будет. Мы настраивали сервер с нуля, поэтому сразу создали необходимые группы на английском языке и установили их соответствие системным UNIX-группам с помощью команд:

```
net groupmap add ntgroup="Domain Admins" unixgroup=ntadmins rid=512 type=d
net groupmap add ntgroup="Domain Users"  unixgroup=ntusers  rid=513 type=d
net groupmap add ntgroup="Domain Guests" unixgroup=nobody rid=514 type=d
net groupmap add ntgroup="Domain Computers" unixgroup=computers type=d
```

Если же оставить название групп на русском, то процесс миграции не будет выполнен корректно — пользователи не будут перенесены на Linux-сервер.

Теперь нужно подготовить Linux-сервер. Нужно убедиться, что на вашем сервере нет групп со следующими GID: 512, 513, 514, 515, 544, 548, 550, 552. Дело в том, что сценарий `smbldap-populate` создаст группы именно с этими GID. И если в системе уже будут группы с такими GID, то сценарию `smbldap-populate` не получится создать необходимые группы, следовательно, не получится произвести процесс миграции. Кстати, именно из-за `smbldap-populate` мы и переименовывали группы пользователей на Windows-сервере — этот сценарий создает группы на английском языке.

Сам процесс миграции будет производиться утилитой Windows to Linux Migration Toolkit (`w2lmt`). Скачать эту утилиту можно по адресу:

http://sourceforge.net/projects/w2lmt/

В состав `w2lmt` входят несколько сценариев. Основной сценарий называется `w2lmt-migrate-smbauth`. Именно этот сценарий переносит пользователей с Windows-сервера на LDAP. Для миграции DNS используется сценарий `w2lmt-migrate-dns`, но мы этот сценарий использовать не будем, поскольку будем считать, что DNS-сервер уже работает под управлением UNIX. Для миграции службы Exchange используется сценарий `w2lmt-migrate-directory`. Во многих случаях Exchange не используется, а ActiveDirectory используется только для аутентификации пользователей, следовательно, вам понадобится запустить только один сценарий — `w2lmt-migrate-dns`.

Все готово для миграции, теперь можно приступить к установке программного обеспечения на Linux-сервер. Настройка Linux-сервера будет производиться на примере дистрибутива Debian. В других дистрибутивах будут небольшие отличия, связанные с использованием других менеджеров пакетов. То есть команды установки пакетов в других дистрибутивах будут иными, а форматы файлов останутся такими же, как описано в этой главе.

21.2. Установка программного обеспечения

Установим необходимые пакеты:

```
$ sudo apt-get install slapd
$ sudo apt-get install apache2 phpldapadmin
$ sudo apt-get install samba smbclient smbfs samba-doc
$ sudo apt-get install smbldap-tools ldap-utils
```

При установке `slapd` (это и есть LDAP) вам будет задан ряд вопросов, ответить на которые нужно так:

```
DNS domain name: ltd
Name of organisation: LTD
Admin password: ваш_пароль
```

```
Confirm password: ваш_пароль
Allow LDAP v2: no
```

Вам нужно ввести имя домена, название организации, пароль администратора и запретить использование LDAP версии 2 (для нашей задачи она нам не нужна).

При установке пакетов samba, smbclient, smbfs, samba-doc нужно ответить на вопросы так:

```
Domain Name: ltd
Use Password Encryption: Yes
Modify smb.conf to use WINS settings via DHCP: No
How to run Samba: daemons
Create password database: Yes
```

21.3. Настройка установленных пакетов

После установки пакетов приступим к настройке. Первым делом нужно скопировать файл /usr/share/doc/samba-doc/example/LDAP/samba.schema.gz в каталог /etc/ldap/schema/:

```
$ sudo cp  /usr/share/doc/samba-doc/example/LDAP/samba.schema.gz /etc/ldap/schema/
```

После этого откройте файл /etc/ldap/slapd.conf и в конец блока # Schema and objectClass definitions добавьте строку:

```
include /etc/ldap/schema/samba.schema
```

У вас должно получиться примерно так:

```
# Schema and objectClass definitions
include /etc/ldap/schema/core.schema
include /etc/ldap/schema/cosine.schema
include /etc/ldap/schema/nis.schema
include /etc/ldap/schema/inetorgperson.schema
include /etc/ldap/schema/samba.schema
```

Полный листинг /etc/ldap/slapd.conf приводить не стану, поскольку в этом нет никакой необходимости. Чтобы проверить, не допустили ли вы ошибку при редактировании этого файла, введите команду:

```
$ slaptest
```

В ответ вы должны получить вот эту строку (означает, что с файлом все в порядке):

```
config file testing succeeded
```

Что же касается файла конфигурации Samba (/etc/samba/smb.conf), то имеет смысл рассмотреть его полностью (листинг 21.1).

Листинг 21.1. Файл /etc/samba/smb.conf

```
# Глобальные параметры
[global]
# Отключаем WINS
wins support = no
```

```
protocol = NT1
name resolve order = lmhosts host bcast
dns proxy = no

# Задаем имя домена, описание и имя Linux-сервера
workgroup = LTD
server string = My LDAP Linux Server
netbios name = LSERVER
netbios aliases = lserver

# Определяем кодировки
dos charset = CP866
unix charset = UTF-8
display charset = UTF-8

# Тип безопасности
security = users
map acl inherit = yes
inherit acls = yes
inherit permissions = yes

# Шифровать пароли, разрешить использовать пустые пароли
encrypt passwords = yes
null passwords = yes

# Включить привилегии
enable privileges = yes
passdb backend = ldapsam:ldap://127.0.0.1
idmap backend = ldap:ldap://127.0.0.1

# Суффиксы LDAP
ldap admin dn = cn=admin,dc=mydomain
ldap suffix = dc=mydomain
ldap machine suffix = ou=Computers
ldap user suffix = ou=Users
ldap group suffix = ou=Groups
ldap passwd sync = yes
ldap delete dn = yes
ldap passwd sync = yes
ldap ssl = no

# Предпочитаемый мастер, разрешен вход в домен
preferred master = yes
domain master = no
domain logons = yes

# Перемещаемых профилей не будет
logon path =
logon drive = Z:
logon home = \\%L\SYSVOL\users\%U

# Сценарии добавления/удаления компьютеров/пользователей
# Обратите внимание: эти сценарии отличаются от аналогичных из
# главы 20
add machine script = /usr/sbin/smbldap-useradd -t 0 -w "%u"
add user script = /usr/sbin/smbldap-useradd -m "%u"
```

продолжение ⇩

Листинг 21.1 *(продолжение)*

```
delete user script = /usr/sbin/smbldap-userdel "%u"
add group script = /usr/sbin/smbldap-groupadd -p "%g"
delete group script = /usr/sbin/smbldap-groupdel "%g"
add user to group script = /usr/sbin/smbldap-groupmod -m "%u" "%g"
delete user from group script = /usr/sbin/smbldap-groupmod -x "%u" "%g"
set primary group script = /usr/sbin/smbldap-usermod -g '%g' '%u'

# Интерфейс
interfaces = eth0
# Наша сеть
hosts allow = 127.0.0.1 192.168.1.0/255.255.255.0
hosts deny = ALL
map to guest = Bad User
guest account = nobody
log file = /var/log/samba/log.%m
max log size = 500 # in Kb
log level = 5
# Оптимизируем Samba
socket options = TCP_NODELAY SO_SNDBUF=8192 SO_RCVBUF=8192
socket address = 127.0.0.1 192.168.1.2

# Поддержка печати CUPS, загружать принтеры
printing = cups
load printers = yes
panic action = /usr/share/samba/panic-action %d

# Стандартные общие ресурсы
[homes]
comment = Home Directories
guest ok = No
browseable = No
create mask = 0644
directory mask = 0775
writeable = yes
inherit permissions = Yes

[profiles]
path = /home/SYSVOL/profiles
comment = User Profiles
guest ok = Yes
browseable = No
create mask = 0600
directory mask = 0700
writeable = Yes
profile acls = Yes
csc policy = disable
force user = %U
valid users = %U @"Domain Admins"

[netlogon]
comment = User Netlogon
path = /home/SYSVOL/netlogon
browseable = No
write list = MYDOMAIN@"Domain Admins"
```

```
[IPC$]
path = /tmp
hosts allow = 192.168.1.0/24, 127.0.0.1
hosts deny = 0.0.0.0/0
```

Конфигурационный файл «готов к употреблению». Вам нужно только изменить имя домена, указать IP-адреса вашей сети. Пока мы настроили наш сервер как резервный контроллер домена:

```
preferred master = yes
domain master = no
domain logons = yes
```

Когда процесс миграции будет завершен, выключите Windows-сервер и установите значение:

```
domain master = yes
```

Параметр `logon path` не установлен, поскольку профили пользователей не будут храниться на сервере. Значение по умолчанию для этого параметра выглядит так:

```
logon path = \\%N\%U\profile
```

Запустите Samba:

```
$ sudo /etc/init.d/samba start
```

Далее нужно установить пароль администратора каталога LDAP:

```
$ sudo smbpasswd -w пароль
```

После этого Samba нужно остановить:

```
$ sudo /etc/init.d/samba stop
```

21.4. Пакеты аутентификации

Если вы внимательно читали главу, то, наверное, заметили, что мы установили не все пакеты из списка заявленных. Нам нужно еще установить следующие пакеты:

```
$ sudo apt-get install libnss-ldap
$ sudo apt-get install libpam-ldap
$ sudo apt-get install nscd
```

Все эти пакеты относятся к пакетам аутентификации. При установке первого пакета вам нужно будет ответить на ряд вопросов:

```
LDAP Server Host: 127.0.0.1
DN of Search Base: dc=mydomain
LDAP Version: 3
Database requires login: no
Make config readable by owner only: yes
```

Установка второго пакета повлечет необходимость ответить на эти вопросы (правильные ответы выделены жирным):

```
Make local root db admin: yes
Database requires logging in : no
```

```
Root login account : cn=admin,dc=mydomain
Root password : ваш_пароль
Crypt : crypt
```

Отредактируем файл /etc/ldap/ldap.conf и приведем его к виду:

```
BASE dc=mydomain
HOST localhost
```

После этого отредактируем файл /etc/nsswitch.conf:

```
passwd: compat ldap
group: compat ldap
shadow: compat ldap
```

Нужно настроить систему для аутентификации через LDAP. Для этого нужно отредактировать файлы /etc/pam.d/common-auth, /etc/pam.d/common-password и /etc/pam.d/common-account.

Начнем с файла /etc/pam.d/common-auth. Нужно найти в нем строку:

```
auth required pam_unix.so nullok_secure
```

Удаляем ее и вместо нее добавляем две строки:

```
auth sufficient pam_ldap.so
auth required pam_unix.so nullok_secure use_first_pass
```

В файле /etc/pam.d/common-password нужно найти и удалить следующую строку:

```
password required pam_unix.so nullok obscure min=4 max=8 md5
```

Вместо нее нужно добавить две строки:

```
password sufficient pam_ldap.so
password required pam_unix.so nullok obscure min=4 max=8 md5 use_first_pass
```

Аналогично, в файле /etc/pam.d/common-account найдем и удалим строку:

```
account required pam_unix.so
```

Вместо нее добавим тоже две строки:

```
account sufficient pam_ldap.so
account required pam_unix.so try_first_pass
```

21.5. Миграция пользователей на LDAP

Скачайте архив w2lmt и распакуйте в любой каталог. Но перед использованием w2lmt нужно отредактировать его конфигурационный файл — migrate-smbauth.conf (листинг 21.2).

Листинг 21.2. Файл migrate-smbauth.conf

```
#source windows PDC host (typically DNS name)
SourceHost="server"
#source windows domain name
SourceDomain="LTD"
#source PDC type, can be NT4,AD
SourceType="NT4"
```

```
#source windows domain administrator account name
SourceAdminAccount="Администратор"
#target linux LDAP server
TargetHost="localhost"
#target linux LDAP port (389 is default)
TargetPort="389"
#location of config file for samba
smbconf="/etc/samba/smb.conf"
#location of config file for smbldap-toolkit
smbldap="/etc/smbldap-tools/smbldap.conf"
```

Самые важные параметры выделены жирным. Параметр SourceHost задает имя Windows-сервера. Параметр SourceDomain — это наш домен.

Тип источника можно выбрать параметром SourceType. Если ActiveDirectory работает в смешанном режиме (mixed mode), нужно установить тип источника NT4. А если ActiveDirectory работает в родном режиме (native mode), нужно установить значение AD.

Почему-то после установки smbldap-tools не устанавливаются конфигурационные файлы /etc/smbldap-tools/smbldap.conf и /etc/smbldap-tools/smbldap_bind.conf. Эти файлы не нужны для перехода на LDAP, вполне уместны параметры по умолчанию, но нужно создать эти файлы, поскольку они необходимы для скриптов миграции (иначе будет ошибка выполнения скрипта), поэтому просто создадим эти файлы:

```
# touch /etc/smbldap-tools/smbldap.conf
# touch /etc/smbldap-tools/smbldap_bind.conf
```

Сценарий w2lmt-migrate-smbauth позволит изменить самые необходимые параметры, но не все. При желании можно заполнить файлы конфигурации /etc/smbldap-tools/smbldap.conf и /etc/smbldap-tools/smbldap_bind.conf. Примеры этих файлов вы найдете в каталоге /usr/share/doc/smbldap-tools/examples. Не забудьте только отредактировать эти файлы.

Для нашей конфигурации эти файлы представлены в листингах 21.3 и 21.4. Самые важные параметры файла /etc/smbldap-tools/smbldap.conf выделены жирным.

Листинг 21.3. Файл /etc/smbldap-tools/smbldap.conf

```
# Подчиненный LDAP-сервер
# Если такового нет, используйте значение "127.0.0.1"
slaveLDAP="127.0.0.1"
# Порт подчиненного LDAP-сервера
# По умолчанию используется значение "389"
slavePort="389"
# Главный LDAP-сервер: нужен для операций записи
# Значение по умолчанию "127.0.0.1"
masterLDAP="127.0.0.1"
# Порт главного LDAP-сервера
# Значение по умолчанию: "389"
masterPort="389"
# Не используем шифрование (TLS), для включения
# TLS установите значение 1
ldapTLS="0"
```

продолжение ⇩

Листинг 21.3 *(продолжение)*

```
# LDAP суффикс
suffix="dc=ltd"
# Где будем хранить информацию о пользователях
usersdn="ou=Users,dc=ltd"
# Где будем хранить информацию о компьютерах
computersdn="ou=Computers,dc=ltd"
# Где будем хранить информацию о группах
groupsdn="ou=Groups,dc=ltd"
# Где будем хранить следующие идентификаторы пользователя и группы
# (uidNumber и gidNumber)
sambaUnixIdPooldn="cn=NextFreeUnixId,dc=ltd"
scope="sub"
# Тип шифрования UNIX (CRYPT, MD5, SMD5, SSHA, SHA, CLEARTEXT)
hash_encrypt="CRYPT"
# Если hash_encrypt = CRYPT, вы должны установить salt_format
# Значение по умолчанию "%s", но многие системы генерируют
# пароли в MD5, если использовать значение "$1$%.8s"
# Параметр не является обязательным
crypt_salt_format="%s"
########################################################################
# Конфигурация учетных записей пользователей UNIX
#
########################################################################
# Оболочка по умолчанию
userLoginShell="/bin/bash"
# Домашний каталог
userHome="/home/SYSVOL/users/%U"
# Права доступа для домашнего каталога
userHomeDirectoryMode="700"
# Gecos
userGecos="Domain User"
# Пользователь по умолчанию (POSIX and Samba) GID
defaultUserGid="513"
# Компьютер по умолчанию (Samba) GID
defaultComputerGid="515"
# Каталог "скелета"
skeletonDir="/etc/skel"
# Возраст пароля в днях
defaultMaxPasswordAge="45"
########################################################################
# Настройка SAMBA
#
########################################################################
# The UNC path to home drives location (%U username substitution)
# Just set it to a null string if you want to use the smb.conf 'logon home'
# directive and/or disable roaming profiles
userSmbHome="\\LAN2\%U"
# Не используем профили пользователей
userProfile=""
# Используем профили:
# userProfile="\\LAN2\profiles\%U"
# Буква диска по умолчанию (для домашнего каталога)
userHomeDrive="Z:"
```

```
# Logon-сценарий по умолчанию
userScript="login.bat"
# Почтовый домен
mailDomain="my.mail.domain"
############################################################################ Конфигурация
SMBLDAP-TOOLS #
############################################################################
with_smbpasswd="0"
smbpasswd="/usr/bin/smbpasswd"
with_slappasswd="1"
slappasswd="/usr/sbin/slappasswd"
```

Листинг 21.4. Файл /etc/smbldap-tools/smbldap_bind.conf

```
slaveDN="cn=admin,dc=ltd"
slavePw="ваш пароль"
masterDN="cn=admin,dc=ltd"
masterPw="ваш пароль"
```

Все готово для миграции сервера на UNIX. Перед миграцией убедитесь, что Samba остановлена, а сервер `slapd` — запущен. Также зайдите по адресу:

```
http://ваш_Linux_сервер/phpldapadmin
```

Для входа выберите **Login**, в поле **Login DN** введите `cn=admin,dc=ltd` и введите пароль, указанный в **/etc/smbldap-tools/smbldap_bind.conf**. Просмотрите, отсутствует ли в базе LDAP запись **sambaDomainName=LTD**. Если запись есть, ее нужно удалить.

Запустим сценарий миграции:

```
$ cd /путь/к/w2lmt
$ sudo ./w2lmt-migrate-smbauth -f /путь/к/migrate-smbauth.conf
```

Сценарий выведет параметры миграции и предложит выбор:

```
Press (C) to continue, (M) to modify the above or (Q) to quit:
```

Если с параметрами все в порядке, нажмите **C**, а если нет, тогда нужно нажать **M** для их изменения. Весь процесс миграции займет 20–30 минут, после чего нужно выключить Windows-сервер и запустить Samba (сделав его контроллером домена; см. выше).

Настройка DHCP-сервера 22

22.1. Назначение DHCP-сервера

Как мы помним, раньше компьютеры стоили очень дорого. Самая дешевая конфигурация IBM PC в 1981 году стоила 1565 долларов, но IBM PC даже не знал, что такое сеть, а вот более мощные компьютеры, которые могли работать в сети, стоили на порядок дороже. Цена в 20 000 долларов за такой компьютер никого удивляла, но позволить себе его мог далеко не каждый. Цена взята не с потолка — в начале 1980-х годов столько стоил IBM 5100. Понятно, что при такой стоимости несколько компьютеров — непозволительная роскошь. Даже если кто-то и мог себе позволить два таких компьютера, то они не нуждались в протоколе автоматической настройки сети.

Но с появлением того же IBM PC компьютеры подешевели и стали ближе к массам. Да, около 2000 долларов за IBM PC в хорошей комплектации для 1981 года — огромная сумма. Но это не 20 000. А к концу 1980-х компьютеры стали еще более доступными. Сетью из нескольких компьютеров уже никого не удивишь, а особо крупные организации могли себе позволить парк из нескольких десятков компьютеров. А теперь подумайте: одно дело настроить два компьютера, другое дело — 50 компьютеров. Поэтому появилась необходимость в протоколе динамической настройки узла. Такой протокол был принят в 1993 году и назван DHCP (Dynamic Host Configuration Protocol — протокол динамической настройки узла).

Кому интересны технические подробности DHCP, рекомендую прочитать RFC 2131 (http://tools.ietf.org/html/rfc2131), а мы рассмотрим общую концепцию протокола DHCP. Итак, у нас есть сеть компьютеров. Сеть может быть абсолютно любой — это может быть и локальная Ethernet-сеть, и беспроводная сеть Wi-Fi, и сеть с сервером удаленного доступа (компьютеры клиентов обычно подключаются по протоколу PPP). В этой сети есть DHCP-сервер. Обычно программа-сервер устанавливается на основном компьютере для этого типа сети. Например, в локальной Ethernet-сети программа-сервер может быть установлена на шлюзе, в Wi-Fi-сети — на точке доступа, в сети интернет-провайдера — на сервере

удаленного доступа. Если позволяют ресурсы, для DHCP-сервера может быть отведен отдельный физический компьютер, но, как правило, это нерационально.

Когда клиент подключается к сети (например, вы подключили физически сетевой кабель к сетевому адаптеру), его операционная система, при условии, что сетевой адаптер не настроен, начинает искать DHCP-сервер в сети.

DHCP-сервер можно обнаружить путем широковещательной рассылки — адрес 255.255.255.255, протокол UDP, порт 68. Клиент производит такую рассылку, чтобы найти сервер.

DCHP-сервер, «увидев» нового клиента, назначает ему все основные сетевые параметры — IP-адрес клиента, IP-адрес шлюза, IP-адреса DNS-серверов, маску сети, имя узла и т. д. После этого клиент может полноценно работать в сети.

Откуда сервер знает, какой адрес нужно присвоить тому или иному компьютеру? Существует две методики назначения IP-адресов: без привязки к MAC-адресу сетевого адаптера и с таковой привязкой. В первом случае в конфигурационном файле сервера администратор указывает диапазон IP-адресов, который должен быть назначен клиентам сети. Рекомендуется указывать диапазон с небольшим запасом. Например, вы можете назначить диапазон адресов 192.168.1.10–100. Когда к сети подключится первый клиент, сервер назначит ему адрес 192.168.1.10, второму — 192.168.1.11 и т. д. У IP-адреса есть время аренды. Время аренды определяется в конфигурационном файле, обычно оно составляет 24 часа. По истечении этого времени сервер «забирает» IP-адрес, помещает его в пул свободных адресов, а клиенту выделяет новый адрес из пула свободных. Может оказаться, что клиенту будет выдан тот же IP-адрес — такая ситуация не исключена. Если компьютер клиента выключен раньше, чем через 24 часа с момента получения IP-адреса, то его IP-адрес автоматически «освобождается» и помещается в пул свободных адресов. Все это позволяет экономить адреса. Представьте, что вы — небольшой интернет-провайдер и вам выделена сеть класса C — 254 адреса. Как минимум два адреса будут зарезервированы для внутренних серверов, остальные можно предоставить клиентам — 252 адреса. Однако у вас может быть больше клиентов, скажем, 350. Но, учитывая, что никогда все 350 клиентов не будут работать одновременно, 252 адреса вам вполне хватит. И в этом вам поможет DHCP-сервер. А ведь если бы адреса назначались вручную, вам бы никогда не хватило этого диапазона.

Хотя сейчас, учитывая дефицит IP-адресов, небольшим провайдерам выделяется гораздо меньше реальных адресов, скажем, всего 8 или 16. Все они используются для внутренних нужд (серверы доступа, DNS-серверы, прокси, серверы для хостинга и т. д.), а клиентам назначаются так называемые локальные адреса, которые благодаря NAT могут взаимодействовать с серверами Интернета. О NAT в UNIX мы поговорим в главе 28. А пока рассмотрим вторую методику назначения IP-адресов DHCP-сервером.

Вторая методика заключается в привязке к MAC-адресу — к аппаратному адресу компьютера. В конфигурационном файле сервера четко прописывается, какому компьютеру какой IP-адрес присвоить. Понятно, что ни о какой экономии речи быть не может, но такая техника может с успехом использоваться для разграничения доступа.

Представим реальную ситуацию. Вы захотели поиграть в домашние сети и стать мега-провайдером для своего дома. Ваш компьютер будет выполнять роль шлюза для всех остальных компьютеров сети. Стоимость затрат минимальна — компьютер, коммутатор с нужным количеством портов и устройство для доступа к Интернету (это может быть ADSL-модем или набор оборудования RadioEthernet). Некоторые пользователи согласились платить вам за неограниченный доступ к Интернету на протяжении всего дня, а некоторые пользователи более экономны, и поскольку с 8 до 19 они находятся на работе, то согласились купить так называемый «Вечерний пакет» — с 19 вечера до 7 утра. Далее все просто. Вы назначаете первой группе адреса из сети 192.168.1.0, а другим — из сети 192.168.2.0. С помощью межсетевого экрана (брандмауэра) или прокси-сервера (смотря как вы решили ограничить доступ) вы блокируете доступ к Интернету второй группе днем. Вот и все. Простейшая система разграничения доступа создана. И не нужно настраивать ни сервер аутентификации, ни пароли, ни VPN-соединения. Все гениальное просто.

Вы можете заметить, что такая система ограничения доступа «дырява» как швейцарский сыр. Да, так и есть. Во-первых, можно самому назначить IP-адрес, а можно и изменить MAC-адрес, например в Linux MAC-адрес сетевого адаптера меняется командами:

```
ifconfig ethN down
ifconfig ethN hw ether <MAC-адрес>
ifconfig up
```

Здесь ethN — это имя сетевого интерфейса. А во FreeBSD/OpenBSD вместо второй команды используется вот эта команда:

```
ifconfig ethN lladdr <mac-address>
```

Но если вы не будете трубить об этом на каждом углу, никто так и не догадается. Да, система не идеальна, но ведь она на то и простейшая. Но такая система подойдет как дополнительный барьер в системе безопасности вашей сети. Особенно актуально такое решение для беспроводных сетей, где доступ к сети будет разрешен компьютерам с определенными MAC-адресами.

Однако такая схема неудобна тем, что вам придется чуть ли не вручную переписать MAC-адреса всех компьютеров сети. Как видите, везде есть свои преимущества и недостатки.

22.2. Установка сервера

Для установки сервера в Linux нужно установить пакет dhcp (или dhcp3-server в Ubuntu). Как это сделать, зависит от используемого вами дистрибутива. Установка программного обеспечения в Linux была рассмотрена в главе 11.

Во FreeBSD нужно установить пакет isc-dhcp3-server. Но если у вас есть рабочая станция, работающая под управлением FreeBSD, то для использования DHCP ее, в отличие от рабочих станций под управлением Linux, нужно настроить дополнительно. На ней нужно установить пакет dhcpclient. Затем нужно открыть /etc/rc.conf. В нем нужно найти строку, начинающуюся с ifconfig, например:

```
ifconfig_em0=…
```

Эту строку нужно изменить так:

```
ifconfig_em="DHCP"
```

После этого выполните сценарий /etc/netstart или перезагрузите компьютер для перезапуска сети.

22.3. Настройка сервера

Синтаксис конфигурационного файла во FreeBSD и Linux не отличается. Отличается только месторасположение файла. В Linux файл конфигурации находится в каталоге /etc, а во FreeBSD — /usr/local/etc/. Название файла (одинаковое в обеих операционных системах) — dhcpd.conf. В качестве примера вы можете использовать или листинги из этой книги или файл примера /usr/local/etc/dhcp.conf.sample (или /usr/share/doc/dhcp-<версия>/dhcpd.conf.sample в Linux).

В новых версиях dhcpd файл конфигурации должен начинаться с директивы ddns-update-style, которая задает тип взаимодействия с DNS-сервером. Если вы не укажете эту директиву, при запуске демона dhcpd (напомню, что в UNIX программа-сервер называется демоном) получите сообщение об ошибке. Но вы также можете получить сообщение об ошибке, если укажете неправильное значение этой директивы. Если у вас вообще нет DNS-сервера (вы его пока не настроили), укажите значение none:

```
ddns-update-style none;
```

Если же в вашей сети есть DNS-сервер, тогда можете использовать одно из значений:

- ah-doc — непосредственное обновление (более перспективная схема взаимодействия DNS и DHCP);
- interim — предварительное взаимодействие DNS и DHCP.

Первая схема еще даже окончательно не утверждена, но рекомендуется использовать именно ее. Сторонникам старого и проверенного нужно использовать вторую схему. Добавьте в начало конфигурационного файла одну из строк:

```
ddns-update-style ah-doc;
ddns-update-style interim;
```

После ddns-update-style следует директива subnet, описывающая вашу сеть. Дабы не описывать каждую директиву отдельно, давайте рассмотрим пример описания реальной подсети с моими комментариями:

```
# Подсеть 192.168.3.0, маска сети 255.255.255.0 (сеть класса C)
subnet 192.168.3.0 netmask 255.255.255.0 {

# Шлюз по умолчанию
option routers 192.168.3.1;

# Маска нашей сети 255.255.255.0
option subnet-mask 255.255.255.0;
```

```
# Домен по умолчанию
option domain-name "domain.ru";

# Широковещательный адрес
option broadcast-address 192.168.3.255;

# Включение IP-Forwarding. Эта опция нужна не всегда
# Вы должны понимать, что делаете
# option ip-forwarding on;

# IP-адреса DNS-серверов сети
option domain-name-servers 192.168.3.1, 192.168.3.2;

# Диапазон IP-адресов для клиентов
# 192.168.3.20 - 192.168.3.100
range 192.168.3.20 192.168.3.100;

# срок аренды адреса - 8 часов - 28 800 секунд
default-lease-time 28800;

# забрать адрес самому через 32 400 секунд (9 часов)
# если клиент не вернул адрес сам через 8 часов, он будет
# возвращен автоматически через 32 400 секунд
max-lease-time 32400;

}
```

Вот практически и все. А вы ожидали длинный файл на несколько страниц? На самом деле основная задача выполнена. Но может оказаться, что у вас несколько подсетей, тогда все подсети (все директивы subnet — по одной для каждой подсети) должны быть описаны «внутри» директивы shared-networks. Простейший пример конфигурации на несколько подсетей выглядит так:

```
shared-network firma {
# Глобальные для всех подсетей параметры
# Наш домен
    option domain-name                  "company.com";
# Наши серверы DNS
    option domain-name-servers          192.168.1.1, 192.168.1.2;
# шлюз по умолчанию
    option routers                      192.168.1.1;

# описываем подсети 192.168.1.0, 192.168.2.0 и 192.168.3.0

    subnet 192.168.1.0 netmask 255.255.252.0 {
        range 192.168.1.5 192.168.1.254;
    }
    subnet 192.168.2.0 netmask 255.255.252.0 {
        range 192.168.2.5 192.168.2.254;
    }
    subnet 192.168.3.0 netmask 255.255.252.0 {
        range 192.168.3.5 192.168.3.254;
    }

}
# * The End *
```

После этого можно указать дополнительные параметры для каждой подсети (если это нужно).

Ранее была рассмотрена альтернативная схема настройки DHCP-сервера — с привязкой к MAC-адресу. Она реализуется путем добавления секции host для каждого компьютера в секцию subnet:

```
subnet 192.168.1.0 netmask 255.255.255.0 {
    host comp_name {
        hardware ethernet xx:xx:xx:xx:xx:xx;
    fixed-address IP-адрес;
    }
}
```

Вместо xx:xx:xx:xx:xx:xx нужно указать MAC-адрес сетевого адаптера, а в качестве значения параметра fixed-address — фиксированный IP-адрес. После служебного слова host указывается имя компьютера.

22.4. Управление сервером

Для запуска и останова DHCP-сервера во FreeBSD используются команды

```
# /usr/local/etc/rc.d/isc-dhcp.sh start
# /usr/local/etc/rc.d/isc-dhcp.sh stop
```

В Linux принято использовать команду service:

```
service dhcpd start
service dhcpd restart
service dhcpd stop
```

Для автоматического запуска DHCP-сервера во FreeBSD откройте /etc/rc.conf и добавьте в него строку:

```
dhcpd_enable="YES"
```

В Linux демон dhcpd автоматически настраивается на автоматический запуск.

Настройка DNS-сервера 23

23.1. Принцип работы системы доменных имен

Прежде чем приступить к настройке DNS-сервера, нужно разобраться, как работает система доменных имен — DNS (Domain Name System). Система DNS используется для преобразования IP-адресов в соответствующие им символьные имена узлов и обратно, то есть для преобразования (разрешения) доменных (символьных) имен в IP-адреса. Управлять компьютерными сетями гораздо проще, если для адресации используются числовые адреса. Но человеку гораздо проще запомнить символьное имя — все числа не запомнишь. Вот поэтому и появилась необходимость в системе DNS.

Пользователь вводит в строке браузера (указывает в почтовом клиенте, в командной строке и т. д.) символьное имя, например www.piter.com. Чтобы установить соединение с удаленным компьютером, сетевой программе (в данном случае браузеру) нужно знать IP-адрес удаленного узла. Браузер обращается к операционной системе с просьбой преобразовать доменное имя в IP-адрес. Программа-резольвер (это часть операционной системы, отвечающая за разрешения доменных имен) обращается к файлу /etc/hosts и пытается там найти адрес удаленного узла. В большинстве случаев она там ничего не находит, поэтому обращается к серверу имен, который указан в файле /etc/resolv.conf.

DNS-сервер, получив запрос от программы-резольвера, пытается найти ответ, то есть IP-адрес, в своем локальном кэше. Если пользователи уже обращались к узлу www.piter.com, в локальном кэше будет адрес этого узла, и он будет передан программе-резольверу. В свою очередь, резольвер передаст IP-адрес браузеру, а тот установит соединение.

Совсем иная ситуация, если доменное имя в кэше не найдено. Тогда сервер обращается к корневому серверу DNS. Корневых серверов много, но у каждого DNS-сервера есть список корневых серверов, к которым можно обратиться. Понятно, что корневой сервер не знает адрес удаленного узла, зато он знает адрес сервера, который поддерживает домен .com. Наш DNS-сервер обращается к этому узлу с просьбой разрешить имя, если и этот узел не знает ответ, то запрос пере-

дается DNS-серверу, обслуживающему домен piter.com, который и возвращает IP-адрес узла www.piter.com.

Примерно так выглядит процесс разрешения доменного имени в IP-адрес. На практике все немного сложнее, но в общих чертах все выглядит именно так. Если вы внимательно прочитали предыдущий абзац, то догадались, что система DNS является иерархической. Корневые DNS-серверы обычно обозначаются точкой (.), затем следуют серверы, обслуживающие домены первого уровня (.ru, .com, .net и др.), потом идет второй уровень доменов (.com.ru, .net.ua, .com.ua и т. д.), затем третий уровень и т. д.

Вы сейчас стоите перед выбором:

- Использовать DNS-сервер провайдера — все просто, не нужно ничего настраивать, достаточно «прописать» адреса DNS-серверов в /etc/resolv.conf и указать их в настройках DHCP-сервера, чтобы эти адреса были автоматически переданы всем остальным узлам сети.
- Настроить собственный полноценный DNS-сервер — имеет смысл, только если у вас крупная сеть, которая нуждается в поддержке собственного домена. Ведь гораздо дешевле заплатить провайдеру за поддержку доменного имени — всего от 5 долларов в год, зато DNS-серверу не придется работать круглосуточно, чем вы сэкономите на оплате электричества (и износе самого сервера).
- Настроить кэширующий DNS-сервер — а вот это решение подойдет для любой сети. Кэширующий DNS-сервер будет собирать локальный кэш IP-адресов, то есть работать подобно серверу провайдера. Но, учитывая, что наш сервер будет находиться в нашей сети и его будут загружать только наши клиенты, работать он будет на порядок быстрее сервера провайдера.

В этой главе мы настроим оба типа DNS-серверов.

23.2. Установка DNS-сервера

В UNIX используется DNS-сервер BIND (Berkley Internet Name Domain). В Linux для установки DNS-сервера нужно установить пакет bind9, а во FreeBSD (если DNS-сервер еще не установлен) — порт net/bind9.

В Linux сервер сразу же настраивается на автоматический запуск, а вот во FreeBSD нужно добавить следующую строку в /etc/rc.conf:

```
named_enable="YES"
```

Обратите внимание: что пакет (порт) называется bind9, а сервис — named. Такая нестыковка наблюдается как во FreeBSD, так и в Linux.

Для управления сервером можно использовать следующие команды:

```
# ndc <start|restart|stop>
# service named <start|restart|stop>
# /etc/rc.d/named <start|restart|stop>
```

Первая команда универсальная — ее можно использовать как в UNIX, так и в Linux. Linux-пользователи больше привыкли к команде service, а вот FreeBSD-пользователи могут управлять сервером с помощью сценария /etc/rc.d/named.

Пока запускать сервер не нужно — мы его не настроили. Лучше отредактируйте файл /etc/resolv.conf на сервере так:

```
search ваш_домен
nameserver 127.0.0.1
nameserver <IP-адрес сервера DNS вашего провайдера>
```

На остальных UNIX-узлах можно отредактировать файл /etc/resolv.conf так:

```
search ваш_домен
nameserver <IP-адрес компьютера, на котором установлена программа named>
```

Хотя правильнее на этом же компьютере развернуть DHCP-сервер и в его настройках указать, какой DNS-сервер нужно использовать пользователям сети. Так будет проще, чем редактировать /etc/resolv.conf на каждом компьютере.

23.3. Конфигурационные файлы сервера

23.3.1. Основной конфигурационный файл named.conf

Наш DNS-сервер будет обслуживать так называемые доменные зоны. На первый взгляд зона очень похожа на сам домен. Сейчас вы поймете почему. Информация о доменных именах и IP-адресах не хранится на одном DNS-сервер, часто используются два сервера — первичный и вторичный. Каждая группа серверов (первичный и вторичный) обслуживают свою часть пространства имен. Это и есть зона. А домен — понятие более абстрактное, его нельзя «пощупать». В то время как зона — это реальный текстовый файл, в котором указана информация о какой-то части пространства имен. Данный текстовый файл хранится на DNS-сервере, точнее на группе серверов.

Вот еще один пример. Пусть у нас есть один корневой сервер. Он обслуживает, скажем, 5 узлов. Но со временем узлов стало 5000, и обращаться с таким количеством узлов в одном едином домене не очень удобно. Поэтому на нашем сервере появились домены первого уровня. Скажем, в каждом домене по 500 узлов. Но 500 узлов — тоже много, поэтому в доменах появились поддомены — домены второго уровня. Потом число узлов выросло до 50 000, количество доменов тоже увеличилось в 10 раз. Часть доменов «переехала» на другие DNS-серверы, чтобы снизить нагрузку на основной сервер. Какие домены целесообразно передать на другие серверы? Правильно, лучше для каждого домена первого уровня создать свой сервер. Но на этих серверах вскоре появилось много поддоменов, поэтому для каждого поддомена тоже был отведен новый сервер. Когда поддомен передается на обслуживание другому серверу, его поддерево (информация об узлах и о доменах 3-го уровня) перемещается на другой сервер. При этом сервер домена более высокого уровня уже не обслуживает вырезанное поддерево. Представили? Исходя из всего этого, домен — это все поддерево со всеми возможными поддоменами, а зона — только часть дерева, без веток (поддоменов).

Конфигурационные файлы DNS-сервера хранятся в каталоге /etc/namedb (/var/named/etc/namedb) во FreeBSD или в каталоге /etc/bind/ в Linux. Хотя имя

каталога зависит от версии BIND и от разработчиков дистрибутива. В некоторых версиях Linux конфигурационные файлы тоже можно найти в каталоге /etc/named.

Особое внимание обратите на файлы named.conf (основной конфигурационный файл) и named.root (список корневых серверов). Листинг файла named.root я приводить не стану, поскольку он у всех один. Скажу только, что его нужно постоянно поддерживать в актуальном состоянии. Для обновления named.root введите команды:

```
# cd /etc/namedb
# fetch ftp://ftp.internic.net/domain/named.root
```

А вот файл named.conf нужно изучить. В листинге 23.1 представлен этот файл с моей машины под управлением FreeBSD 8. В Linux синтаксис этого файла будет таким же. Комментарии, как обычно, на русском языке, но часть комментариев, мало относящихся к делу, я удалил для сокращения листинга. Наиболее важные комментарии, после которых следуют параметры конфигурационного файла и на которые вам следует обратить внимание, я выделил жирным.

Листинг 23.1. Файл /etc/named/named.conf

```
// $FreeBSD: src/etc/namedb/named.conf,v 1.29.2.1 2009/08/03 08:13:06 kensmith Exp $
//
// Подробную информацию вы сможете найти в документации из каталога
// /usr/share/doc/bind9
//

// Примеры комментариев:
// Комментарий, тип 1 (в стиле C)
# Комментарий, тип 2 (в стиле bash)
/* Комментарий, тип 3 (для многострочных комментариев) */

options {
    // Глобальные параметры сервера,
    // пути указаны относительно chroot-каталога
    // Главный каталог
    directory    "/etc/namedb";
    // PID-файл демона DNS-сервера
    pid-file    "/var/run/named/pid";
    // dump-файл (используется для отладки ошибок)
    dump-file    "/var/dump/named_dump.db";
    // файл статистики
    statistics-file    "/var/stats/named.stats";

// Определяет IP-адрес интерфейса, на котором будет работать сервер
// Если вы настраиваете кэширующий сервер, укажите 127.0.0.1
// При настройке реального сервера нужно указать и реальный адрес
// или вообще закомментировать директиву listen-on – тогда сервер
// будет работать на всех сетевых интерфейсах
    listen-on    { 127.0.0.1; };

// Поддержка IPv6. Поскольку IPv6 обычно не используется,
// директиву listen-on-v6 лучше закомментировать
//    listen-on-v6    { ::1; };
```

продолжение ⇩

Листинг 23.1 *(продолжение)*

```
// Если вы планируете удалить связанные пустые зоны (см. ниже),
// то закомментируйте эти строки
    disable-empty-zone "255.255.255.255.IN-ADDR.ARPA";
    disable-empty-zone "0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.IP6.
    ARPA";
    disable-empty-zone "1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.IP6.
    ARPA";

// **** ВАЖНЫЙ БЛОК ****
// Блок forwarders содержит DNS-серверы провайдера. Наш сервер будет
// получать кэш имен от этих серверов, что снизит наш трафик и уменьшит
// время разрешения доменных имен
/*
    forwarders {
        127.0.0.1;
    };
*/

// При настройке "ленивого" сервера, который должен только передавать
// запросы серверам, указанным в forwarders, но никогда не порождать
// собственные запросы, раскомментируйте строку:
//    forward only;

// Раскомментируйте строку ниже для включения автоматического
// перенаправления, использующего записи из /etc/resolv.conf
// Также нужно добавить строку named_auto_forward=yes в /etc/rc.conf
//    include "/etc/namedb/auto_forward.conf";

/*
Современные версии BIND могут использовать случайные UDP-порты
для каждого исходящего запроса. Это позволяет защитить сервер от атаки, но требует
дополнительной настройки брандмауэра. Подробно можно прочитать здесь:
http://en.wikipedia.org/wiki/DNS_cache_poisoning

NNNNN нужно заменить числом в диапазоне 49160 – 65530.
*/
// query-source address * port NNNNN;
};

// При настройке локального сервера имен добавьте nameserver 127.0.0.1 в ваш
// файл /etc/resolv.conf и убедитесь, что вы включили сервер в rc.conf

// Имя файла, содержащего список корневых серверов имен.
zone "." { type hint; file "named.root"; };

/* ВНИМАНИЕ! Далее следует описание зон, поддержка которых
   снизит нежелательный трафик, ускорит процесс преобразования имен. Если
   вы не желаете поддерживать эти зоны, удалите строки, начиная с этого
   комментария до комментария END_CUT
   А вот если вы хотите включить поддержку этих зон, раскомментируйте
   их описание. Текст между слэш-звездочка и звездочка-слэш считается
   комментарием */

/*
```

```
zone "." {
    type slave;
    file "slave/root.slave";
    masters {
        192.5.5.241;    // F.ROOT-SERVERS.NET.
    };
    notify no;
};
zone "arpa" {
    type slave;
    file "slave/arpa.slave";
    masters {
        192.5.5.241;    // F.ROOT-SERVERS.NET.
    };
    notify no;
};
zone "in-addr.arpa" {
    type slave;
    file "slave/in-addr.arpa.slave";
    masters {
        192.5.5.241;    // F.ROOT-SERVERS.NET.
    };
    notify no;
};
*/

// END_CUT

// RFC 1912: не удаляйте эти зоны
zone "localhost"    { type master; file "master/localhost-forward.db"; };
zone "127.in-addr.arpa" { type master; file "master/localhost-reverse.db"; };
zone "255.in-addr.arpa"    { type master; file "master/empty.db"; };

// Далее можно удалить все, что связано с IPv6. В данном листинге
// я решил этого не делать, чтобы листинг менее отличался
// от вашего файла named.conf

// RFC 1912 для IPv6
zone "0.ip6.arpa"    { type master; file "master/localhost-reverse.db"; };

// "данная" сеть (RFC 1912 и 3330)
zone "0.in-addr.arpa"        { type master; file "master/empty.db"; };

// Частные сети (RFC 1918)
zone "10.in-addr.arpa"        { type master; file "master/empty.db"; };
zone "16.172.in-addr.arpa"    { type master; file "master/empty.db"; };
zone "17.172.in-addr.arpa"    { type master; file "master/empty.db"; };
zone "18.172.in-addr.arpa"    { type master; file "master/empty.db"; };
zone "19.172.in-addr.arpa"    { type master; file "master/empty.db"; };
zone "20.172.in-addr.arpa"    { type master; file "master/empty.db"; };
zone "21.172.in-addr.arpa"    { type master; file "master/empty.db"; };
zone "22.172.in-addr.arpa"    { type master; file "master/empty.db"; };
zone "23.172.in-addr.arpa"    { type master; file "master/empty.db"; };
zone "24.172.in-addr.arpa"    { type master; file "master/empty.db"; };
zone "25.172.in-addr.arpa"    { type master; file "master/empty.db"; };
zone "26.172.in-addr.arpa"    { type master; file "master/empty.db"; };
```

продолжение ⇩

Листинг 23.1 *(продолжение)*

```
zone "27.172.in-addr.arpa"    { type master; file "master/empty.db"; };
zone "28.172.in-addr.arpa"    { type master; file "master/empty.db"; };
zone "29.172.in-addr.arpa"    { type master; file "master/empty.db"; };
zone "30.172.in-addr.arpa"    { type master; file "master/empty.db"; };
zone "31.172.in-addr.arpa"    { type master; file "master/empty.db"; };
zone "168.192.in-addr.arpa"   { type master; file "master/empty.db"; };

// RFCs 3330 и 3927
zone "254.169.in-addr.arpa"   { type master; file "master/empty.db"; };

// сеть для тестирования (RFC 3330)
zone "2.0.192.in-addr.arpa"   { type master; file "master/empty.db"; };

// RFC 3330 (тестирование производительности роутера)
zone "18.198.in-addr.arpa"    { type master; file "master/empty.db"; };
zone "19.198.in-addr.arpa"    { type master; file "master/empty.db"; };

// Зарезервировано IANA - старый класс E
zone "240.in-addr.arpa"        { type master; file "master/empty.db"; };
zone "241.in-addr.arpa"        { type master; file "master/empty.db"; };
zone "242.in-addr.arpa"        { type master; file "master/empty.db"; };
zone "243.in-addr.arpa"        { type master; file "master/empty.db"; };
zone "244.in-addr.arpa"        { type master; file "master/empty.db"; };
zone "245.in-addr.arpa"        { type master; file "master/empty.db"; };
zone "246.in-addr.arpa"        { type master; file "master/empty.db"; };
zone "247.in-addr.arpa"        { type master; file "master/empty.db"; };
zone "248.in-addr.arpa"        { type master; file "master/empty.db"; };
zone "249.in-addr.arpa"        { type master; file "master/empty.db"; };
zone "250.in-addr.arpa"        { type master; file "master/empty.db"; };
zone "251.in-addr.arpa"        { type master; file "master/empty.db"; };
zone "252.in-addr.arpa"        { type master; file "master/empty.db"; };
zone "253.in-addr.arpa"        { type master; file "master/empty.db"; };
zone "254.in-addr.arpa"        { type master; file "master/empty.db"; };

// Неназначаемые адрес IPv6 – можно удалять (RFC 4291)
zone "1.ip6.arpa"      { type master; file "master/empty.db"; };
zone "3.ip6.arpa"      { type master; file "master/empty.db"; };
zone "4.ip6.arpa"      { type master; file "master/empty.db"; };
zone "5.ip6.arpa"      { type master; file "master/empty.db"; };
zone "6.ip6.arpa"      { type master; file "master/empty.db"; };
zone "7.ip6.arpa"      { type master; file "master/empty.db"; };
zone "8.ip6.arpa"      { type master; file "master/empty.db"; };
zone "9.ip6.arpa"      { type master; file "master/empty.db"; };
zone "a.ip6.arpa"      { type master; file "master/empty.db"; };
zone "b.ip6.arpa"      { type master; file "master/empty.db"; };
zone "c.ip6.arpa"      { type master; file "master/empty.db"; };
zone "d.ip6.arpa"      { type master; file "master/empty.db"; };
zone "e.ip6.arpa"      { type master; file "master/empty.db"; };
zone "0.f.ip6.arpa"    { type master; file "master/empty.db"; };
zone "1.f.ip6.arpa"    { type master; file "master/empty.db"; };
zone "2.f.ip6.arpa"    { type master; file "master/empty.db"; };
zone "3.f.ip6.arpa"    { type master; file "master/empty.db"; };
zone "4.f.ip6.arpa"    { type master; file "master/empty.db"; };
zone "5.f.ip6.arpa"    { type master; file "master/empty.db"; };
zone "6.f.ip6.arpa"    { type master; file "master/empty.db"; };
zone "7.f.ip6.arpa"    { type master; file "master/empty.db"; };
```

```
zone "8.f.ip6.arpa"    { type master; file "master/empty.db"; };
zone "9.f.ip6.arpa"    { type master; file "master/empty.db"; };
zone "a.f.ip6.arpa"    { type master; file "master/empty.db"; };
zone "b.f.ip6.arpa"    { type master; file "master/empty.db"; };
zone "0.e.f.ip6.arpa"  { type master; file "master/empty.db"; };
zone "1.e.f.ip6.arpa"  { type master; file "master/empty.db"; };
zone "2.e.f.ip6.arpa"  { type master; file "master/empty.db"; };
zone "3.e.f.ip6.arpa"  { type master; file "master/empty.db"; };
zone "4.e.f.ip6.arpa"  { type master; file "master/empty.db"; };
zone "5.e.f.ip6.arpa"  { type master; file "master/empty.db"; };
zone "6.e.f.ip6.arpa"  { type master; file "master/empty.db"; };
zone "7.e.f.ip6.arpa"  { type master; file "master/empty.db"; };

// IPv6 ULA – можно удалять (RFC 4193)
zone "c.f.ip6.arpa"        { type master; file "master/empty.db"; };
zone "d.f.ip6.arpa"        { type master; file "master/empty.db"; };

// IPv6 локальная связь – можно удалять (RFC 4291)
zone "8.e.f.ip6.arpa"        { type master; file "master/empty.db"; };
zone "9.e.f.ip6.arpa"        { type master; file "master/empty.db"; };
zone "a.e.f.ip6.arpa"        { type master; file "master/empty.db"; };
zone "b.e.f.ip6.arpa"        { type master; file "master/empty.db"; };

// IPv6 запрещенные адреса – тоже можно удалить (RFC 3879)
zone "c.e.f.ip6.arpa"        { type master; file "master/empty.db"; };
zone "d.e.f.ip6.arpa"        { type master; file "master/empty.db"; };
zone "e.e.f.ip6.arpa"        { type master; file "master/empty.db"; };
zone "f.e.f.ip6.arpa"        { type master; file "master/empty.db"; };

// IP6.INT – запрещено – можно удалить (RFC 4159)
zone "ip6.int"            { type master; file "master/empty.db"; };

// Ниже описываются ваши зоны

/* Пример динамической зоны
key "exampleorgkey" {
    algorithm hmac-md5;
    secret "секретный код – у вас он другой";
};
// Впишите сюда ваш домен и раскомментируйте код
zone "example.org" {
    type master;
    allow-update {
        key "exampleorgkey";
    };
    file "dynamic/example.org";
};
*/

/* Подчиненная обратная зона
zone "1.168.192.in-addr.arpa" {
    type master;
    file "slave/1.168.192.in-addr.arpa";

};
*/
```

Файл named.conf выглядит просто ужасающе. Либо вы испугаетесь, либо вам просто надоест его читать. Я решил вам немного помочь и «почистил» примерный named.conf. Результат представлен в листинге 23.2 — сокращенный named.conf.

Листинг 23.2. Файл named.conf (сокращенный вариант)

```
options {
    // главный каталог каталог
    directory    "/etc/namedb";
    // PID-файл демона
    pid-file    "/var/run/named/pid";
    // dump-файл (для отладки)
    dump-file    "/var/dump/named_dump.db";
    // файл статистики
    statistics-file    "/var/stats/named.stats";

// Блок forwarders (чуть позже мы его опишем подробно)
/*
    forwarders {
        127.0.0.1;
    };
*/

// Имя файла со списком корневых серверов имен.
zone "." { type hint; file "named.root"; };

// RFC 1912:не нужно удалять эти зоны
zone "localhost"    { type master; file "master/localhost-forward.db"; };
zone "127.in-addr.arpa" { type master; file "master/localhost-reverse.db"; };
zone "255.in-addr.arpa"    { type master; file "master/empty.db"; };

// "данная" сеть (RFC 1912 и 3330)
zone "0.in-addr.arpa"        { type master; file "master/empty.db"; };


// Ниже описываются ваши зоны

/* Пример зоны
key "exampleorgkey" {
    algorithm hmac-md5;
    secret " секретный код — у вас он другой";
};
// Когда расскомментируете пример, сюда нужно вписать ваш домен
zone "example.org" {
    type master;
    allow-update {
        key "exampleorgkey";
    };
    file "dynamic/example.org";
};
*/

/* Пример подчиненной (slave) обратной зоны
zone "1.0.10.in-addr.arpa" {
    type master;
```

```
    file "slave/1.0.10.in-addr.arpa";
};
*/
```

Можно пойти дальше и сократить named.conf до варианта, представленного в листинге 23.3. На этот раз я удалил даже комментарии, но назначение директив и блоков вы можете узнать из листинга 23.2.

Листинг 23.3. Файл named.conf (вариант 3)

```
options {
directory "/etc/namedb";
};
zone "example.com" {
type master;
file "master/example.com";
};
zone "1.0.10.in-addr.arpa" {
type master;
file "master/1.0.10.in-addr.arpa";
};
zone "." {
type hint;
file "named.root";
};
zone "0.0.127.in-addr.arpa" {
type master;
file "master/localhost.rev";
};
zone "0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.IP6.INT" {
type master;
file "master/localhost.rev";
};
```

В листинге 23.3 представлен самый простой, но тем не менее вполне рабочий вариант. После изменения файла конфигурации нужно перезапустить DNS-сервер, чтобы изменения вступили в силу.

Я обещал рассказать о блоке forwarders. В этом блоке указываются серверы-экспедиторы, к которым наш сервер будет обращаться с просьбой разрешить доменное имя (или IP-адрес). Использование экспедиторов позволит снизить нагрузку на корневые серверы имен и уменьшить время разрешения узла. Серверами-экспедиторами могут выступать серверы провайдера. Вот пример заполнения блока forwarders:

```
forwarders {
10.0.0.1;
10.0.0.2;
};
```

23.3.2. Файлы зон

Поддерживаемые нашим сервером зоны описываются в файле name.conf так:

```
zone "firma.com" {
type master;
```

```
file "master/firma.com";
}
```

Это самое простое описание зоны. Параметр file задает имя файла зоны, а параметр type — тип зоны (табл. 23.1).

Таблица 23.1. Тип зоны

Тип зоны	Описание
master	Главная зона. Означает, что наш сервер является главным сервером для этой зоны и на нем хранится главный файл описания зоны. Все изменения в этом файле производятся собственноручно вами — администратором

slave	Подчиненная зона. Сервер хранит копию зоны, которую он получает от главного (master) сервера. Изменение реплики зоны осуществляется автоматически — через определенный промежуток времени копия зоны приходится от главного (первичного, primary) сервера. Пример описания подчиненной зоны:
	zone "firma.com" {
	type slave;
	file "slave/firma.com";
	masters { 10.0.1.1; };
	};
	В блоке masters указываются IP-адреса серверов, содержащих мастер-копию зоны. Полученная информация сохраняется в файле <главный каталог>/slave/firma.com (например, /etc/namedb/slave/firma.com)

hint	Этот тип используется только для корневой зоны
stub	Похож на slave, но передает только NS-записи DNS, а slave передает все записи (A, MX, NS и т. д.)
forward	Применяется для пересылки запросов на другой сервер, который указан в блоке forwarders

Настоятельно рекомендую ограничить круг узлов, с которыми может «общаться» наш сервер. Директива allow-query определяет, кто может использовать наш сервер. Ответ очевиден — только узлы нашей сети. Директива allow-transfer определяет узлы, которые могут получить копию зоны, то есть в директиве allow-transfer нужно указать все подчиненные (вторичные, secondary) серверы DNS. Пример:

```
type master;
file "firma.com";
allow-query { 10.10.1/24; localhost; };
allow-transfer { 10.0.1.2; };
};
```

Осталось самое малое: создать файл зоны. В нашем случае файл будет называться master/firma.com (листинг 23.4).

Листинг 23.4. Файл master/firma.com

```
$TTL 7200
firma.com. IN SOA ns1.firma.com. admin.firma.com. (
1000000000          ; Серийный номер
10800               ; Обновить
1000                ; Повторить
604800              ; Не годен
```

```
86400 )            ; TTL
; Наши серверы DNS
@             IN NS ns1.firma.com.
@             IN NS ns2.firma.com.

; Определяем имена узлов
localhost     IN A 127.0.0.1
ns1           IN A 10.0.1.1
ns2           IN A 10.0.1.2
den           IN A 10.0.1.3
mail          IN A 10.0.1.4
@             IN A 10.0.1.1

; Псевдонимы узлов
www           IN CNAME @

; Запись MX (почтовик)
@             IN MX 10 mail.firma.com.
```

Файл только кажется простым. В нем нужно разобраться. В первой строке указывается «время жизни» зоны — 7200 секунд. Для большой и часто изменяющейся зоны данное значение нужно уменьшить. После этого начинается запись SOA (Start Of Authority) — это основная запись зоны. После названия зоны/домена (firma.com) указывается первый сервер DNS, затем — адрес администратора сервера, но в адресе вместо @ пишется точка. После имени сервера DNS и e-mail администратора обязательно нужно ставить точку, иначе после строки будет добавлено имя домена (в нашем случае firma.com). Согласитесь, адрес admin@firma.com.firma.com — это не то, что нам нужно.

Параметры (обновить, повторить и др.) можно не изменять, но разобраться с ними нужно:

- Серийный номер — используется для определения актуальности файла зоны (у вас серийный номер будет другой). После каждого изменения файла вы должны увеличивать это число на единицу. Если у вас нет вторичного сервера DNS, то этот параметр изменять необязательно.
- Обновить — частота обновления файла зоны. У нас каждые 10 800 секунд вторичный DNS-сервер будет получать файл зоны от первичного сервера.
- Повторить — если не получилось по какой-то причине обновить файл зоны, вторичный сервер предпримет повторную попытку через 1000 секунд.
- Не годен — если через 604 800 секунд не будет получен мастер-файл, вторичный сервер объявляет зону «не пригодной к употреблению» и сбрасывает всю информацию о зоне.
- TTL — определяет, сколько секунд нужно хранить ответы «негативного кэша».

Как видите, файл небольшой, но довольно важный. И это только начало. А далее описываются наши серверы DNS. Обычно их два — первичный и вторичный:

```
@             IN NS ns1.firma.com.
@             IN NS ns2.firma.com.
```

Собственно, тут ничего сложного нет, только не забудьте о точке. Если у вас начальная стадия склероза, вообще не указывайте строку «firma.com», например:

```
@            IN NS ns1
@            IN NS ns2
```

Запись NS, как вы уже догадались, используется для определения DNS-серверов зоны. А вот запись A используется для преобразования доменных имен в IP-адреса:

```
localhost    IN A 127.0.0.1
ns1          IN A 10.0.1.1
ns2          IN A 10.0.1.2
den          IN A 10.0.1.3
mail         IN A 10.0.1.4
@            IN A 10.0.1.1
```

Символ @ очень важен. Когда пользователь попытается разрешить имя firma.com без указания узла, то будет передан IP-адрес первичного DNS-сервера. Удобно, чтобы на этом сервере был запущен веб-сервер. Тогда при вводе адреса http://firma.com будет открыт сайт компании, а не получено сообщение об ошибке.

ПРИМЕЧАНИЕ

В этой главе, как вы уже успели заметить, мы используем только локальные IP-адреса. На практике, скорее всего, у вас будут реальные IP-адреса.

Узел с именем www создавать необязательно. Мы лучше создадим псевдоним для этого доменного имени:

```
www          IN CNAME @
```

Когда пользователь в строке браузера введет URL http://www.firma.com, он получит IP-адрес первого DNS-сервера, на котором, как мы договорились, запущен веб-сервер — 10.0.1.1.

Запись MX определяет почтовые серверы домена:

```
@         IN MX 10 mail.firma.com.
```

Число после MX — это приоритет сервера. Чем ниже число, тем важнее сервер. У вас может быть несколько почтовых серверов, главным будет сервер с самым низким приоритетом.

С файлом firma.com мы успешно разобрались. Но это только файл описания прямой зоны (используется для преобразования доменных имен в IP-адреса), а ведь еще есть и обратная зона, которая отвечает за преобразование IP-адресов в доменные имена — обратное преобразование. Файл обратной зоны тоже задается в named.conf:

```
zone "1.0.10.in-addr.arpa" {
type master;
file "master/1.0.10.in-addr.arpa";
};
```

Файл обратной зоны называется master/1.0.10.in-addr.arpa. Вообще не имеет значения, как вы его назовете, главное самому не запутаться и знать, что и где

«лежит». А вот описывать зону нужно так: "1.0.10.in-addr.arpa" — по этому имени сервер понимает, что мы описываем обратную зону. В данном случае зона соответствует подсети 10.0.1.0. Последний 0 в имени зоны мы не указываем.

В листинге 23.5 представлен файл обратной зоны.

Листинг 23.5. Файл master/1.0.10.in-addr.arpa

```
$TTL 7200
1.0.10.in-addr.arpa. IN SOA ns1.firma.com. admin.firma.com. (
100000000             ; Серийный номер
10800                 ; Обновить
1000                  ; Повторить
604800                ; Не годен
86400 )               ; TTL

@     IN NS ns1.firma.com.
@     IN NS ns1.firma.com.

1     IN PTR ns1.firma.com.
2     IN PTR ns2.firma.com.
3     IN PTR den.firma.com
4     IN PTR mail.firma.com.
```

Здесь все просто. Заголовок похож на файл прямой зоны, а вместо записей A используются записи PTR. Записи CNAME и MX в этом файле не нужны. IP-адрес тоже можно не указывать полностью. Если вы указываете IP-адрес полностью, то не забывайте его указывать в обратном порядке:

```
3.0.0.10    IN    PTR    den.firma.com
```

23.4. Настройка кэширующего DNS-сервера

Зачем нужен кэширующий DNS-сервер, вы уже знаете. Но зачем его настраивать, если особой необходимости в нем нет. Ведь разрешение доменного имени занимает всего одну секунду и даже меньше. Да, вы правы, когда вы вводите доменное имя в строке браузера, то практически мгновенно система разрешает это имя. Потом браузер загружает страницу, а в ней — ссылки на объекты, скажем картинки, баннеры, которые находятся на других серверах. Таких объектов может быть несколько десятков. Выходит, системе нужно разрешить еще несколько десятков доменных имен. Вот поэтому есть смысл настроить кэширующий сервер— вы сэкономите как свое время, так и трафик.

Настроить кэширующий сервер DNS совсем не сложно. В листинге 23.6 приведен файл named.conf кэширующего сервера. Можете использовать этот файл, а можете немного изменить его — по своему усмотрению.

Листинг 23.6. Файл named.conf кэширующего сервера DNS

```
options {
directory "/etc/namedb";
};
```

```
forward first;
forwarders {
// указываем здесь IP-адреса DNS-серверов провайдера
}

zone "." {
type hint;
file "named.root";
};

zone "0.0.127.in-addr.arpa" in {
type master;
file "master/localhost.rev";
};

zone "0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.IP6.INT" {
type master;
file "master/localhost.rev";
};
```

Можно еще в этот файл добавить следующие строки:

```
logging {
category lame-servers { null; };
category cname { null; };
};
```

Они избавят вас от надоедливых предупреждений в журналах системы.

Как видно из листинга, кэширующий сервер не обслуживает никаких зон. Но файл named.conf — это не единственный файл, который вам необходимо отредактировать. Вам придется изменить файл обратной зоны для localhost (листинг 23.7).

Листинг 23.7. Файл обратной зоны для localhost (localhost.rev)

```
$TTL 7200
@ IN SOA ns1.firma.com. root.firma.com. (
1000000000         ;
3600             ;
900             ;
3600000         ;
3600 )             ;

IN     NS ns1.firma.com.
1     IN PTR localhost.firma.com.
```

Вот и все. А вы думали.

Прокси-сервер 24

24.1. Прокси-сервер Squid

24.1.1. Назначение прокси-сервера

Основное назначение прокси-сервера — кэширование веб-страниц. Поскольку страницы, загруженные через прокси, кэшируются, это ускоряет повторный доступ к ним (потому что передача страницы осуществляется уже с локального прокси, а не с удаленного сервера) и экономится интернет-трафик (уже не нужно обращение к удаленному серверу).

Пусть один из пользователей локальной сети запросил страницу www.dkws.org.ua/index.php. На ней много картинок и текста, а канал, по современным меркам, да еще и для предприятия, — узкий, всего 1 Мбит/с, поэтому страница загрузилась не так быстро, как нам бы этого хотелось. После первого обращения к странице она помещается в кэш прокси-сервера — кэшируется. Все последующие обращения к странице приводят к ее загрузке из кэша прокси, а не с удаленного сервера. Скорость передачи данных по локальной сети — 100 Мбит/с, но никак не 1 Мбит/с, поэтому страница загрузится намного быстрее. Выходит, что мы не только ускорили доступ к Интернету, но еще и сэкономили деньги — все последующие обращения к странице не будут включены в счет за оплату услуг провайдера, потому что мы работаем с локальной копией страницы, которая доступна всем пользователям сети.

Кроме кэширования страниц, Squid можно использовать для запрещения доступа к узлам, содержащим «контент для взрослых», запретить показ баннеров (что тоже сэкономит трафик), запретить доступ к Интернету определенных пользователей.

24.1.2. Установка Squid

В Linux прокси можно установить с помощью менеджера пакетов — нужно установить пакет squid. Во FreeBSD можно воспользоваться как утилитой pkg_add, так и установкой из портов.

```
# cd /usr/ports/www/squid
# make install clean
```

Чтобы потом не забыть, сразу добавьте в /etc/rc.conf строку автоматического запуска прокси:

```
squid_enable="YES"
```

24.1.3. Конфигурационный файл прокси

Во FreeBSD конфигурационный файл называется /usr/local/squid/etc/squid.conf. В Linux — /etc/squid/squid.conf. Формат конфигурационного файла в Linux и FreeBSD одинаковый. Но при редактирования конфигурационного файла обращайте внимание на пути к разным файлам (например, к протоколам) — во FreeBSD и в Linux они могут отличаться. Поэтому если у вас есть рабочий прокси под FreeBSD, нельзя просто скопировать его конфигурационный файл на сервер под управлением Linux и надеяться, что все будет работать.

Полностью приводить конфигурационный файл squid.conf я не буду. Это и не нужно — для нормальной работы вам необходимо отредактировать несколько директив. Начнем с директив управления кэшем — это самые важные директивы:

```
cache_peer proxy.isp.ru
cache_mem 256 MB
cache_dir /usr/local/squid/cache 2048 16 256
cache_swap_high 97
cache_swap_low 85
maximum_object_size 10240 KB
minimum_object_size 0 KB
```

Первая директива по умолчанию закомментирована. Она позволяет указать прокси-сервер провайдера, чтобы ваш сервер стал «соседом» этого прокси, — тогда кэш можно будет «черпать» от вашего соседа. Перед тем как включать этот параметр, посоветуйтесь с технической поддержкой вашего провайдера.

Вторая директива устанавливает размер кэша в оперативной памяти. В данном случае мы установили размер кэша в 256 Мбайт. Для небольшой сети этого вполне достаточно. Третья директива устанавливает имя каталога, где будет храниться кэш (убедитесь, чтобы этот каталог принадлежал пользователю squid), размер кэша (2048 Мб), количество каталогов первого уровня (16), количество каталогов второго уровня (256).

Директива cache_swap_high устанавливает размер кэша в процентном соотношении, при достижении которого начнется усиленный выброс кэша из оперативной памяти на жесткий диск. Старые объекты будут помещены на жесткий диск, освобождая место для новых объектов. Директива cache_swap_low задает размер кэша в процентном соотношении, когда прекращается процесс усиленного свопа кэша.

Последние две директивы задают соответственно максимальный и минимальный размеры объекта. Файлы большего или меньшего размера в кэше не сохраняются.

Теперь определим узлы, которые могут использовать наш прокси, в данном случае прокси-сервер могут использовать все узлы сети 192.168.2.0:

```
acl allowed_hosts src 192.168.2.0/255.255.255.0
acl localhost src 127.0.0.1/255.255.255.255
```

Разрешаем SSL-порты:

```
acl SSL_ports port 443 563
```

Разрешаем обычные порты (HTTP и FTP):

```
acl allow_ports port 80 21
```

Разрешаем доступ к узлам локальной сети, метод CONNECT для SSL-портов, все остальное запрещаем:

```
http_access deny CONNECT !SSL_ports
http_access deny !allow_ports
http_access allow localhost
http_access allow allowed_hosts
http_access allow SSL_ports
http_access deny all
```

Практически вся настройка уже выполнена. Но нам этого мало. Давайте еще определим узлы, которым разрешен доступ к Интернету:

```
acl allowed_hosts src "/usr/local/squid/hosts.allow"
```

Ранее мы определили `acl` с именем allowed_hosts, разрешающий доступ к Интернету всем узлам из сети 192.168.2.0. Если нужно разрешить доступ не всем узлам, а только определенным, удобнее перечислить их в отдельном файле (в нашем случае /usr/local/squid/hosts.allow) — по одному узлу в строке. Все это делается для вашего удобства, чтобы не засорять лишней информацией конфигурационный файл.

Владельцем файла hosts.allow должен быть пользователь squid или пользователь, от имени которого вы запускаете Squid. Файл hosts.allow может выглядеть примерно так:

```
# Администратор
192.168.2.2/255.255.255.255
# Директор
192.168.2.3/255.255.255.255
# Гл. бухгалтер
192.168.2.4/255.255.255.255
```

Использование такого способа удобно, если нужно предоставить доступ к Интернету не всем узлам сети, а только определенным. Однако у данного решения есть и недостатки — оно не подходит для сети, где используется DHCP-сервер.

Список узлов Интернета, к которым следует запретить доступ, можно задать так:

```
acl adult urlpath_regex "/usr/local/squid/adult.txt"
http_access deny adult
```

Файл adult.txt содержит регулярные выражения. Если имя узла Интернета соответствует этому выражению, то доступ к этому узлу будет запрещен. Вот пример такого файла:

```
^http://www.sex.com
^http://www.porno.com
^http://www.ero.com
…
```

Понятно, что формировать этот файл вручную никому не хочется. Чуть позже мы рассмотрим решение этой проблемы. А пока сохраним файл конфигурации и запустим `squid` с параметром `-z` для обнуления кэша:

```
# squid -z
```

24.1.4. Запуск, перезапуск и останов сервера

Для управления прокси-сервером используются следующие команды:

```
# В FreeBSD
# /usr/local/etc/rc.d/squid start
# /usr/local/etc/rc.d/squid restart
# /usr/local/etc/rc.d/squid stop
# В Linux
# service squid start
# service squid restart
# service squid stop
```

24.1.5. Отказ от баннеров, рекламы и прочего нежелательного контента

Ранее был показан пример, как можно запретить доступ к некоторым узлам Интернета. Пример хорош, если нужно запретить доступ к нескольким популярным сайтам (например, к социальным сетям), чтобы пользователи занимались работой, а не виртуальным общением с другими пользователям. Когда же нужно запретить баннеры, рекламу и прочий нежелательный контент вроде сайтов для взрослых, наш способ не сработает. Только потому, что вы не сможете добавить в файл `adult.txt` все баннерные сети, все порносайты и т. д. Их очень много, да и каждый день появляются все новые и новые сайты. Пусть за нас это сделает кто-то другой.

Идея заключается в установке редиректора Rejik, который будет отсеивать «вредные» сайты на основании базы бан-листов, которая регулярно обновляется. Вам нужно один раз установить Rejik, а затем только следить за базой бан-листов, которую время от времени нужно скачивать в Интернете.

Редиректор Rejik — не единственная программа в своем классе. Можно также использовать программу squidGuard — тоже довольно эффективный способ борьбы с нежелательным контентом. Но описание настройки squidGuard можно без особых проблем найти в Интернете, а вот Rejik — менее популярное средство ограничения контента, но при этом функциональность остается на уровне squidGuard. А учитывая простоту настройки, Rejik стал «героем» этой книги.

Установим Rejik из портов:

```
# cd /usr/ports/www/rejik
# make install
# cp -R /usr/ports/www/rejik/work/banlists /usr/local/rejik
# mkdir /usr/local/www/rejik
# cp -R /usr/ports/www/rejik/work/squid-like-www-en /usr/local/www/rejik
```

Первые две команды устанавливают Rejik, при установке выберите все доступные опции. Третья команда копирует бан-листы в рабочий каталог `rejik`.

Последняя команда нужна, если на компьютере, на котором вы будете запускать Rejik, установлен веб-сервер.

Rejik работает так: если узел является «вредным», то Rejik перенаправляет запрос пользователя на наш локальный сервер, содержащий страницу с сообщением об ошибке/недоступности узла и т. д. Можно сделать перенаправление на любой другой нейтральный узел, например на mail.ru. Но желательно перенаправлять на локальный веб-сервер. Если веб-сервер уже настроен в вашей сети, но на другом компьютере, нужно скопировать содержимое каталога /usr/ports/www/rejik/work/squid-like-www-en в каталог /usr/local/www/rejik вашего веб-сервера.

Подкаталог Rejik обычно не существует, поэтому его нужно создать и предоставить ему права доступа для пользователя, от имени которого у вас запускается Apache.

В файл squid.conf нужно добавить строки:

```
url_rewrite_program        /usr/local/rejik/redirector /usr/local/rejik/rejik.conf
url_rewrite_children       10
```

Этим мы указываем Squid, что нужно использовать программу-редиректор /usr/local/rejik/redirector с конфигурационным файлом /usr/local/rejik/rejik.conf. Сам файл /usr/local/rejik/rejik.conf выглядит так:

```
# Журнал ошибок Rejik
error_log /usr/local/rejik/rejik.err
# Журнал замен: сюда записывают сведения о том, какие страницы
# заменил Rejik
change_log /usr/local/rejik/rejik.log

# Разрешен доступ узлам сети 192.168.2.0
work_ip 127.0.0.1/8
work_ip 192.168.2.0/24

make-cache /usr/local/rejik/make-cache

# Запрещаем баннеры, 192.168.2.1 - адрес вашего веб-сервера
<BANNER>
ban_dir /usr/local/rejik/banlists/banners
url http://192.168.2.1/rejik/1x1.gif
# Баннеры будут заменены файлом 1x1.gif

# Запрещаем порно
<PORNO>
ban_dir  /usr/local/rejik/banlists/porno
url http://192.168.2.1/rejik/porno.html

# Запрещаем скачивать MP3-файлы всем, кроме себя любимого -
# узел с адресом 192.168.2.1 - это узел администратора
<MP3>
ban_dir /usr/local/rejik/banlists/mp3
url http://192.168.2.1/ban/mp3.html
allow_ip 192.168.2.1
```

Подробно файл конфигурации Rejik описан на сайте:

http://rejik.ru/index_ru_4_0.html

Вот, собственно, и все. Перезапускаем Squid и наслаждаемся работой Rejik.

24.1.6. Не забываем настроить клиенты. Прозрачный прокси-сервер

Чтобы все работало как нужно, вы должны соответствующим образом настроить клиенты. В настройках браузера нужно указать IP-адрес нашего прокси-сервера и порт (в нашем случае 3128). Если клиентов много (а на каждом может быть установлено несколько браузеров), то труд нужно проделать большой. Гораздо проще настроить прозрачный прокси-сервер (рис. 24.1), что очень удобно — проще настроить один компьютер, чем 50.

Для этого в squid.conf добавляем строку:

```
http_port 3129 transparent
```

или

```
http_port 3129 intercept
```

Первую строку нужно добавить, если у вас Squid версии 3.0 или младше, вторую — если версия Squid 3.1 или старше. Squid должен быть собран с опциями SQUID_IPFW, SQUID_PF и SQUID_IPFILTER, в зависимости от используемого вами межсетевого экрана.

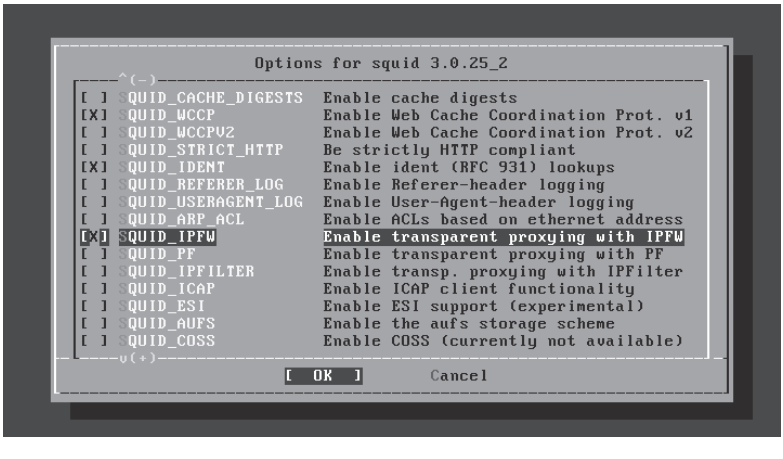

Рис. 24.1. Включение «прозрачного» прокси при сборке

Осталось только в /etc/rc.d/rc.firewall.local добавить строку:

```
add fwd SQUIDIP,3128 tcp from any to any 80 recv IFACE
```

Здесь SQUIDIP — адрес сервера, на котором запущен прокси, а IFACE — интерфейс, откуда поступают запросы клиентов (интерфейс, «смотрящий» в локальную сеть).

На всякий случай отмечу, что для работы прозрачного прокси ваше ядро должно быть собрано со следующими опциями:

```
options         IPFIREWALL
options         IPFIREWALL_FORWARD
options         IPFIREWALL_VERBOSE
options         IPFIREWALL_DEFAULT_TO_ACCEPT
options         IPDIVERT
```

24.2. Прокси-сервер OOPS

Прокси-сервер Squid может показаться кому-то слишком сложным. Именно поэтому в данной книге рассматривается альтернативный прокси OOPS — компактный и быстрый прокси-сервер.

В этой главе OOPS будет сразу настраиваться как прозрачный прокси-сервер. Сначала установим OOPS (при сборке желательно включить опцию PCRE):

```
# cd /usr/ports/www/oops
# make install clean
```

Сразу добавим строку автоматического запуска в /etc/rc.conf:

```
oops_enable="YES"
```

Не забудьте в /etc/rc.d/rc.firewall.local добавить строку (см. выше):

```
add fwd SQUIDIP,3128 tcp from any to any 80 recv IFACE
```

Данное правило брандмауэра сделает его прозрачным. Откройте конфигурационный файл /usr/local/etc/oops/oops.cfg, сделайте прокси прозрачным и определите его «прозрачный» порт:

```
group mynet {
              ...
# Включаем модуль transparent
redir_mods transparent;
             ....
}

module transparent {
    myport 3128
    }
```

Далее определите параметры кэша:

```
storage {
        path /usr/local/oops/storages/oops_storage;
        size 700m;
}
```

Сохраните файл конфигурации и создайте хранилище для кэша:

```
# oops -z
```

Запустите прокси:

```
# /usr/local/etc/rc.d/oops start
```

Компактный и быстрый прокси поддается настройке всего за 10–15 минут. Он выполняет базовые функции прокси и ничего больше. OOPS поддерживает модули редиректоров, что позволяет «привязывать» к нему посторонние программы, но такая конфигурация уже становится сложной, а для сложного прокси-сервера рекомендуется использовать Squid. Все элементарно просто: даже если самостоятельно не получится настроить тот или иной редиректор, ответ можно найти в Интернете. А вот информации по OOPS откровенно мало. Дополнительную информацию по этому прокси вы можете получить на сайте http://oops-cache.org/.

FTP-серверы в UNIX

25

25.1. Перед настройкой сервера

Перед настройкой FTP-сервера нужно знать, зачем он используется, и решить, нужен ли он лично вам. Протокол FTP (File Transfer Protocol) используется для передачи файлов с FTP-сервера и на FTP-сервер. В Интернете FTP-сервер очень популярен, но наша книга ориентирована на предприятие, а нужен ли FTP-сервер в корпоративной среде?

В любом случае в корпоративной среде нужно обмениваться файлами. Файлы можно передать по электронной почте, «расшарить» каталог в Windows-сети, а также использовать FTP-сервер (можно использовать и другие сервисы вроде ICQ, но обмен файлами для них — второстепенная функция). Электронная почта для обмена файлами неудобна — если нужно отправить несколько файлов всем пользователям, то в итоге можно легко забить весь жесткий диск почтового сервера. Представим, что нужно отправить файл объемом 10 Мбайт всем 100 пользователям сети, в итоге на почтовом сервере места станет на 1 Гбайт меньше, потому что копия этого файла будет помещена в почтовый ящик каждого пользователя. А если бы вы поместили этот файл в общий каталог (с помощью Samba) или выложили на FTP-сервере, то он бы занял всего 10 Мбайт, даже если бы его скачало не 100, а 1000 пользователей — размер от этого не меняется. Но это только 10 Мбайт. По сегодняшним меркам — это очень мало. А про файлы еще большего размера речь даже не идет — именно по этому администраторы почтовых серверов устанавливают ограничение на размер письма — не более 10–20 Мбайт.

Выходит, для организации файлового хранилища можно использовать или общую папку Windows (Windows share) или FTP-сервер. Чтобы не «расшаривать» папку на каждом компьютере, можно создать общую папку на контроллере домена — так будет удобнее.

Что же использовать? Если в вашей сети пока еще нет контроллера домена и вы не планируете его использовать, намного проще настроить FTP-сервер и использовать какой-нибудь удобный FTP-клиент. Я рекомендую FileZilla, тем более что есть его версии как для Windows, так и для Linux.

Некоторые администраторы могут со мной поспорить, мол, в Windows уже есть все необходимое программное обеспечение, с которым все умеют работать, — ничего не нужно доустанавливать для организации общих папок. Так то оно так. Но как всегда есть свои преимущества и недостатки. Для полноценной работы механизма общего доступа к ресурсам нужно на каждой рабочей станции запускать соответствующие службы, которые отъедают драгоценные ресурсы, даже если пользователь на своем компьютере не создавал общих папок. Службы запущены на тот случай, что если пользователю вдруг захочется создать общую папку, то ему не придется «дергать» администратора, чтобы тот запустил необходимые службы[1]. К тому же если мы планируем плавный переход на UNIX и свободное программное обеспечение, то FTP — это выход. Ведь для использования общих папок Windows на каждом UNIX-клиенте придется запускать Samba — по сути, инородный сервис для мира UNIX. В случае с FTP запустить сетевой сервис нужно только на одном компьютере — на FTP-сервере. На остальных компьютерах нужно запускать FTP-клиент только при работе с FTP-сервером, но FTP-клиент не будет постоянно «висеть» в памяти и занимать процессорное время.

Что же касается удобства использования, то тут FTP мало чем отличается от механизма общих папок. Вы можете легко организовать файловый обменник — предоставить всем анонимный доступ к общей папке `/pub`, в которой будет все, что вам захочется, — программы, документы, ISO-образы, музыка и даже фильмы (хотя с фильмами это я перегнул, в рабочее время нужно работать, а не фильмы смотреть). А можно создать персональные хранилища файлов для каждого пользователя сети: каждый пользователь сможет хранить свои документы на FTP-сервере и получить к ним доступ с любого компьютера сети, указав персональные имя пользователя и пароль при регистрации на FTP-сервере.

Как уже было отмечено, везде есть свои преимущества и недостатки. Заменив механизм общих папок FTP-сервером, мы получили центральный и более защищенный файловый сервер. Но общие ресурсы Windows — это не только файлы, но и принтеры. А вот управлять принтерами через FTP никак не получится. Тут есть два варианта. Если вы используете только локальные принтеры, то можете смело использовать FTP в качестве основного сервиса обмена файлами. А вот если нужны сетевые принтеры, то придется использовать общие ресурсы Windows или Samba на UNIX-клиенте. Хотя, как показывает практика, на крупных предприятиях можно вообще обойтись без общих ресурсов Windows. Потому как в небольших офисах сетевой принтер — это обычно самый обычный принтер, подключенный к Windows-компьютеру с разрешенной возможностью удаленной печати. То есть, по сути, принтер является сетевым виртуально благодаря программному обеспечению, установленному на компьютере. Такая схема позволяет сэкономить деньги — ведь нужно покупать не принт-сервер, а самый обычный принтер. Но если компьютер, к которому подключен этот принтер, выключен, распечатать документы не получится. В небольшом офисе не проблема подойти к компьютеру и включить его. А вот если офис большой... Тогда, как правило,

[1] В хорошо настроенной и управляемой сети «конечный пользователь» в принципе не должен иметь возможности создавать общие ресурсы, тем более минуя администратора. И это никоим образом не должно зависеть от технологии, реализующей общие ресурсы в данной сети.

покупают принт-серверы. Принт-сервер — это настоящий сетевой принтер, подключаемый к Ethernet-сети непосредственно, а не через компьютер[1]. В этом случае общие ресурсы нам вообще не нужны — все необходимое программное обеспечение будет запущено на принт-сервере. Как видите, FTP при определенных условиях можно использовать как основной сервис обмена файлами.

25.2. Стандартный клиент ftp

Перед настройкой сервера нужно разобраться с использованием FTP-клиента — программы `ftp`. Программа `ftp` — это несколько неудобная (хотя к ней быстро привыкаешь, в отличие от `vi`) текстовая программа. Знать, как использовать программу `ftp`, просто необходимо — ведь графический интерфейс и возможность установить FileZilla есть не всегда.

Для подключения к FTP-серверу используется команда:

```
$ ftp имя-сервера
```

Вот пример подключения к серверу:

```
Connected to ftp.firma.com
220 ProFTPD Server 1.3.2 ready.
Name (ftp.firma.com:denis):ftp1
331 Password required for ftp1.
Password:
230 User ftp1 logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
```

Мы подключились к серверу `ftp.firma.com`, нас поприветствовал сервер ProFTPD версии 1.3.2, мы указали имя пользователя `ftp1` и пароль для этого пользователя. Сервер сообщил нам, что пользователь `ftp1` успешно вошел и сервером используется двоичный режим для передачи файлов. Двоичный режим самый удобный, поскольку подходит как для передачи двоичных, так и текстовых файлов. А вот если передавать в текстовом режиме двоичные (бинарные) файлы, то они будут повреждены.

После регистрации на сервере вы увидите приглашение для ввода FTP-команд:

```
ftp>
```

Введите любую команду, например `ls`, для отображения содержимого каталога:

```
ftp> ls
200 PORT command successful
150 Opening ASCII mode data connection for file list
-rw-r--r-- 1 denis denis 170592 Nov 12 03:50 report.doc
-rw-r--r-- 1 denis denis 988401 May 10 07:21 chap5.doc
-rw-r--r-- 1 denis denis 153677 Oct 15 04:49 toc.xml
```

[1] Впрочем, сейчас существуют и автономные «маленькие» принт-серверы, к которым можно подключить самый обычный (исходно не сетевой) принтер. По цене вполне доступны и для малого офиса, и даже для дома. Вот только часто они несколько «подтормаживают».

```
226 Transfer complete
ftp> quit
221 Goodbye.
```

В домашнем каталоге пользователя три файла: report.doc, chap5.doc и toc.xml. После вывода содержимого каталога мы ввели команду quit для отключения от сервера и выхода из программы ftp. Если нужно только отключиться от сервера, но не завершать работу программы ftp, следует использовать команду close. Список других команд программы ftp представлен в табл. 25.1.

Таблица 25.1. Список команд программы ftp

Команда	Описание
!	Выйти в оболочку
$	Запустить макрос
account ID	Используется для передачи идентификатора пользователя серверу: account <ID пользователя>
append локальный_файл [удаленный_файл]	Добавляет локальный файл к удаленному файлу на сервере. Если имя удаленного файла не указано, считается, что имя такое же, как у локального файла
ascii	Переводит сервер в текстовый (ASCII) режим. Такой режим нельзя использовать для передачи бинарных файлов
bell	Программа ftp будет издавать гудок после каждой выполненной команды (слегка раздражает, подходит разве что при копировании больших файлов — тогда гудки будут реже)
binary	Переводит сервер в бинарный (двоичный) режим передачи данных. Этот режим используется по умолчанию и позволяет передавать двоичные и текстовые файлы
bye	Завершает FTP-соединение и производит выход из программы ftp
case	Переключает регистр символов имен файлов на сервере. Если вы включили этот режим, то все прописные буквы в именах удаленных файлов будут (при копировании файлов командой mget) заменены строчными
cd каталог	Изменить текущий каталог
cdup	Перейти в родительский каталог
chmod права файл	Аналог команды chmod в UNIX — используется для изменения прав доступа к файлу
close	Разрывает соединение, но не завершает работу программы ftp. Открыть новое соединение можно командой open
delete файл	Удаляет файл на сервере (если права доступа к файлу позволяют сделать это)
debug	Включает режим отладки
dir [удаленный_каталог] [локальный_файл]	Выводит содержимое указанного удаленного каталога в локальный файл. Если удаленный каталог не указан, подразумевается текущий удаленный каталог. Если не указан локальный файл, производится вывод на консоль
disconnect	Псевдоним для команды close
exit	Псевдоним для команды bye
get удаленный_файл [локальный_файл]	Загружает удаленный файл в ваш домашний каталог. Файл будет сохранен под именем, заданным вторым параметром. Если же второй параметр не задан, то имя будет такое же, как на сервере

Команда	Описание
hash	Бесполезная опция: перед каждым блоком данных, который отправляется удаленным сервером, будет выводиться знак #
help [команда]	Без параметров позволяет просмотреть список команд, с параметром — вывести справку по указанной в параметре команде
idle	Возвращает и устанавливает таймер простоя на сервере
image	Псевдоним для команды binary
lcd	Позволяет изменить рабочий локальный каталог (чтобы не выходить в оболочку и не вводить команду cd)
ls	Псевдоним для команды dir
macdef	Позволяет определить макрос
mdelete список	Используется для множественного удаления файлов на сервере: mdelete файл1 файл2
mdir список	Множественный вывод содержимого каталогов сервера. Позволяет вывести содержимое сразу нескольких каталогов. Каталоги указываются через пробел
mget список	Множественное копирование файлов с сервера на локальную машину, файлы разделяются пробелами: mget file1 file2 file3
mkdir каталог	Используется для создания каталога на сервере, вы должны обладать соответствующими правами для этой операции
mls	Псевдоним для команды mdir
mode	Изменяет режим передачи файлов (ascii, binary)
modtime файл	Выводит дату последнего изменения удаленного файла
mput список	Множественная загрузка файлов с локального каталога на сервер, имена файлов разделяются пробелами
newer файл	Копирование более нового файла: файл с сервера копируется, только если он новее, чем файл с аналогичным именем в локальном каталоге
open адрес_сервера	Устанавливает соединение с удаленным сервером
prompt	Используется для включения/выключения подтверждения действий при выполнении множественных команд (mget, mput и т. д.)
passive	Переключает сервер в пассивный режим работы; полезно, если брандмауэр блокирует FTP-соединения
sendport	FTP-команда PORT используется при установке соединения для каждой передачи данных. Использование этой команды позволяет предотвратить задержки при передаче нескольких файлов. Но для старых FTP-серверов полезно отключить команду PORT, поскольку она ими не поддерживается. Команда sendport позволяет включить/выключить использование FTP-команды PORT
put локальный_файл [удаленный_файл]	Загружает локальный файл на сервер. Второй параметр можно не указывать — тогда имя файла на сервере будет таким же, как и на локальном компьютере
pwd	Выводит имя текущего каталога на сервере (удаленный рабочий каталог)
quit	Псевдоним для команды bye
recv	Псевдоним для команды get
reget	Используется для докачки файла — загрузка файла начинается с конца локального файла

продолжение ⇩

Таблица 25.1 *(продолжение)*

Команда	Описание
rstatus	Выводит статус сервера
rename старое_имя новое_имя	Используется для переименования файла на сервере
reset	Сброс (очистка) очереди команд
restart	Сброс счетчика передач файлов
rmdir	Удаляет каталог на сервере (при наличии соответствующих прав)
runique	Включает/выключает сохранение файлов с уникальными именами в локальном каталоге
send	Псевдоним для команды put
site	Используется для установки некоторых параметров сервера или получения специальной информации от сервера. Получить подсказку по параметрам можно путем команды rhelp (выполняется на стороне сервера): rhelp site
size файл	Сообщает размер удаленного файла
status	Выводит текущий статус
system	Сообщает тип удаленной системы
sunique	Включает/выключает сохранение файлов с уникальными именами на сервере
trace	Включает/выключает трассировку пакетов
user имя пароль	Производит аутентификацию пользователя
umask	Устанавливает маску создания файлов
verbose	Выключает/выключает вывод подробной информации. Когда эта команда включена, выводится больше информационных (и порой совершенно ненужных) сообщений

Теперь, когда мы знаем, как использовать программу `ftp`, можно приступить к установке FTP-сервера. FTP-сервер, как и любой другой сервер, это обычная программа. А программы бывают разными. Во FreeBSD по умолчанию используется сервер `ftpd`, в Linux раньше использовался сервер `wu-ftpd`, сейчас чаще используется `ProFTPD`. Кроме этих серверов есть еще замечательный FTP-сервер `vsftpd`. Какой сервер использовать — зависит от вас. Если вы раньше администрировали ProFTPD, то наверняка выберете его, поскольку знакомы с его конфигурационным файлом. В этой книге будут рассмотрены два сервера — стандартный `ftpd` (потому что он установлен во FreeBSD по умолчанию) и альтернативный `ProFTPD` (потому что он чаще других используется в Linux).

25.3. Настройка сервера ftpd

25.3.1. Настройка обычного FTP-сервера

Тип сервера и тип запуска сервера

Перед настройкой сервера нужно решить ряд организационных вопросов. Прежде всего нужно выяснить, для чего будет использоваться будущий сервер. Если

только для общего хранилища файлов (фильмов, музыки, программ), то целесообразно настроить анонимный сервер. С таким сервером просто работать — в качестве имени пользователя нужно указать anonymous (или guest), а в качестве пароля — адрес электронной почты. Это за пользователя сделает FTP-клиент. Пользователь же вообще ничего не будет вводить, а сразу получит список файлов и каталогов сервера.

Можно настроить и самый обычный сервер, чтобы каждый пользователь мог хранить собственные файлы в своем домашнем каталоге. По большому счету, можно настроить сервер так, что он будет разрешать хранить файлы в домашних каталогах (если пользователь при регистрации вводит реальные имя пользователя и пароль) и позволять анонимный доступ к файловому хранилищу. В этом случае доступа к домашним каталогам пользователей не будет, но будет доступ к общему каталогу сервера (он обычно называется /pub).

Также нужно выяснить, какой тип запуска сервера вы будете использовать — через inetd или автономный. Автономный тип запуска рекомендуется использовать при больших нагрузках на сервер (если число потенциальных пользователей большое). А вот запуск через inetd рекомендуется использовать, если обращения к серверу будут производиться редко — время от времени.

Обеспечиваем запуск сервера

Итак, приступим к настройке самого обычного FTP-сервера. Сначала обеспечим запуск сервера. Если вы выбрали автономный тип запуска, тогда добавьте в ваш файл /etc/rc.conf следующие строки:

```
ftpd_enable="YES"                    # YES = включить автономный ftpd
ftpd_program="/usr/libexec/ftpd"     # Путь к ftpd
ftpd_flags=""                        # Параметры автономного ftpd
```

Для запуска сервера будет использоваться команда:

```
# /etc/rc.d/ftpd start
```

Можно, конечно, перезагрузить компьютер, но, на мой взгляд, проще ввести приведенную выше команду.

Теперь разберемся, как обеспечить запуск через inetd. Первым делом нужно проверить, а запускается ли вообще inetd? Введите команду:

```
# ps -ax | grep inetd
```

Если в выводе только команда grep inetd, то сервер inetd не запущен. Откройте ваш /etc/rc.conf и добавьте в него строку:

```
inetd_enable="YES"
```

Сохраните файл и введите команду запуска суперсервера inetd:

```
# /etc/rc.d/inetd start
```

Теперь снова убедимся, что inetd запущен:

```
# ps -ax | grep inetd
```

Вывод должен быть примерно таким:

```
966  ??  Is    0:00.01  /usr/sbin/inetd -wW -C 60
989  v0  S+    0:00.01  grep inetd
```

Вот теперь все нормально: суперсервер inetd запущен. Затем откройте файл конфигурации inetd — /etc/inetd.conf и раскомментируйте следующую строку (рис. 25.1):

```
ftp stream tcp nowait root /usr/libexec/ftpd ftpd -l
```

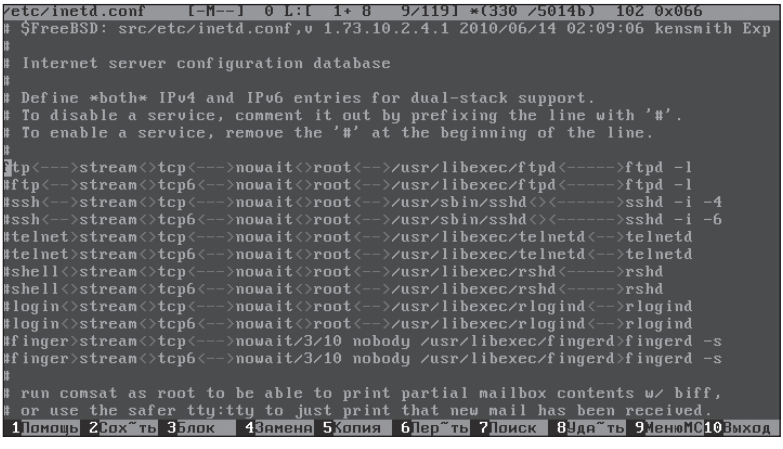

Рис. 25.1. Редактирование файла /etc/inetd.conf редактором mcedit

Сохраните файл и перезапустите inetd:

```
# /etc/rc.d/inetd restart
```

Вообще-то правильнее было бы сначала отредактировать inetd.conf, а потом уже запускать inetd, но теперь вы знаете правильную последовательность действий и сделаете все как нужно.

Теперь нужно проверить работоспособность самого FTP-сервера:

```
# ftp localhost
```

В ответ должны увидеть примерно следующее:

```
Trying 127.0.0.1…
Connected to localhost.
220 denhost.localdomain FTP server (Version 6.00LS) ready.
Name (localhost:root):
```

На рис. 25.2 изображена самая короткая FTP-сессия — я попытался указать имя пользователя root, но FTP-сервер не позволяет регистрацию root, поэтому он сразу завершил соединение с сообщением Login failed.

Нужно отметить, что отклик от сервера, запускаемого в автономном режиме, происходит гораздо быстрее, чем от сервера, запускаемого через inetd. К слову, после сообщения Connected to localhost прошло несколько секунд, прежде чем я увидел приглашение сервера.

Чтобы убедиться, что наш сервер виден другим компьютерам в сети, перейдите за другой компьютер и подключитесь к серверу:

```
ftp адрес_сервера
```

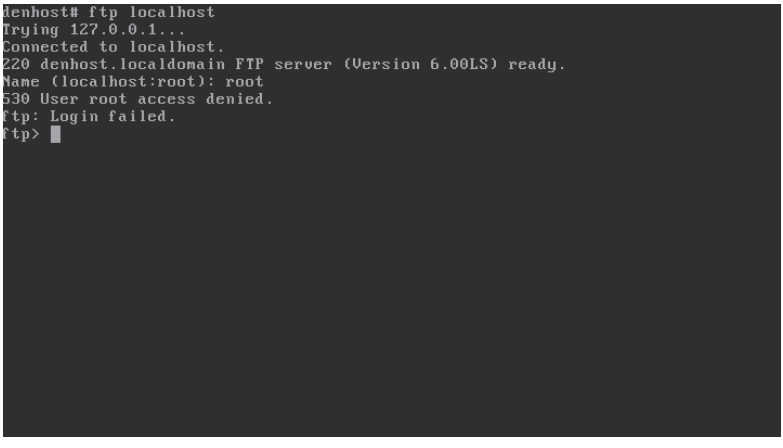

Рис. 25.2. Как root входить нельзя из соображений безопасности

На рис. 25.3 изображен стандартный FTP-клиент Windows (программа `ftp`), подключенный к только что настроенному серверу.

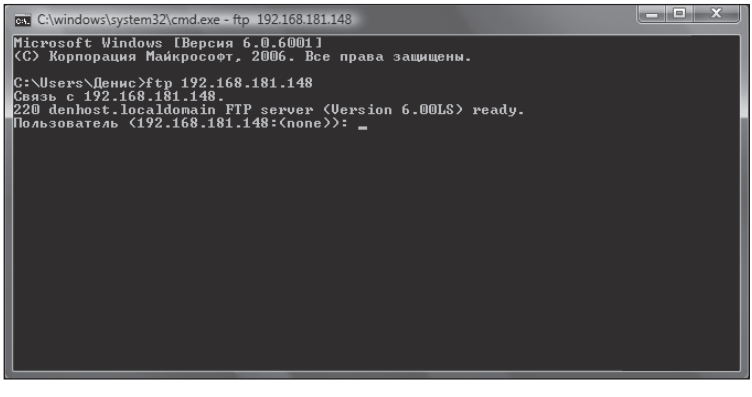

Рис. 25.3. Программа ftp в Windows

Конфигурационные файлы сервера. Пользователи

Сведения о пользователях FTP-сервер получает из общей базы данных пользователей (в Linux — это файлы /etc/passwd и /etc/shadow, в FreeBSD — таблицы, которые строятся на основании файлов /etc/passwd и master.passwd). Другими словами, чтобы добавить FTP-пользователя, просто воспользуйтесь командой `adduser` на FTP-сервере.

Есть одно большое отличие анонимного доступа от обычного (когда указываются реальные имя пользователя и пароль). При анонимном доступе удаленный пользователь получает доступ только к общему каталогу, выйти за пределы этого каталога он не может. При обычном доступе пользователь получает доступ

к корневой файловой системе сервера и может путешествовать по ней, но сохранять файлы может только в своем домашнем каталоге. И это не мудрено — ведь пользователь является самым обычным пользователем системы и имеет на это право. Однако с точки зрения безопасности это не совсем правильно. Рекомендуется всех FTP-пользователей прописать в файле /etc/ftpchroot — тогда корневым для них станет их домашний каталог, и пользователь не сможет выйти за его пределы. А это очень важно. Ведь даже если пользователь не может изменить файлы в каталоге /etc, он может их просмотреть. А в некоторых таких файлах содержатся пароли в незашифрованном виде (например, пароли PPPoE-соединения).

Часть конфигурационных файлов сервера находится в каталоге /etc, а часть — в /var/ftp. Анонимные пользователи не должны видеть файлы и каталоги, находящиеся за пределами каталога /var/ftp, поэтому часть конфигурационных файлов пришлось переместить в этот каталог.

Черный список пользователей, которым запрещается входить по FTP, находится в каталоге /etc/ftpusers. Да, имя файла правильное. В нем действительно содержится черный список FTP-пользователей. Почему у файла такое название (ведь его могли назвать, к примеру, ftpusers.deny, чтобы сразу было понятно, зачем он нужен), я не знаю, так задумали разработчики FTP-сервера, а поскольку я к их числу не отношусь, мое дело маленькое — рассказать вам, зачем используется этот файл.

Итак, в файл /etc/ftpchroot нужно записать пользователей, которым разрешен вход по FTP, а в файл /etc/ftpusers — пользователей, которым вход по FTP запрещен. Пользователи записываются по одному в одной строке, например:

```
root
operator
toor
daemon
bin
…
```

Чтобы занести в этот файл целую группу пользователей (вносить пользователей по одному не удобно, особенно если у вас их много), укажите перед именем группы символ @ — это признак группы:

```
@group1
@group2
```

При попытке зайти на сервер пользователь, находящийся в черном списке, получит сообщение:

```
530 User <имя> access denied.
```

В каталоге /etc вы найдете (если их нет, значит, данные файлы нужно создать) следующие файлы, относящиеся к настройке FTP:

- /etc/ftphosts — содержит конфигурацию виртуальных FTP-узлов;
- /etc/ftpwelcome — содержимое этого файла выводится сразу после подключения пользователя к серверу (файл приветствия);
- /etc/ftpmotd — содержит «сообщения дня», содержимое этого файла выводится после регистрации пользователя на сервере (после того как пользователь указал имя пользователя и пароль).

В каталоге /var/ftp вы найдете следующие файлы и каталоги:

- /var/ftp/bin — содержит программы ls и date. Эти команды используются для вывода содержимого каталога FTP без обращения к системным программам /bin/ls и /bin/date. Все сделано для того, чтобы пользователь никак не вышел за пределы «песочницы» — chroot-окружения.
- /var/ftp/etc — содержит некоторые конфигурационные файлы, эти файлы не имеют отношения к конфигурации вашего реального сервера, они нужны для создания «эффекта наличия» реального сервера. Даже если пользователь их прочитает, ничего страшного не случится. В этом каталоге также есть файл ftpmotd, содержащий приветствие, которое будет показано анонимным пользователям.
- /var/ftp/pub — публичный каталог анонимного сервера, сюда нужно поместить все файлы, которые вы хотите сделать общими, — ISO-образы, документы, музыку и т. д.
- /var/ftp/incoming — необязательный каталог, используется для загрузки файлов на сервер анонимными пользователями.

25.3.2. Настройка анонимного сервера

Теперь настроим анонимный сервер. Запустите конфигуратор sysinstall:

```
# sysinstall
```

Выберите команду Configure ▸ Networking ▸ Anon FTP. В появившемся окне (рис. 25.4) нажмите Yes, а после этого появится окно с настройками анонимного FTP (рис. 25.5).

Обычно данные параметры устраивают всех, поэтому просто нажмите OK. После этого вам будет предложено создать файл приветствия для анонимных пользователей (рис. 25.6). Если вы согласитесь, будет запущен редактор, указанный в переменной окружения EDITOR, для редактирования файла приветствия.

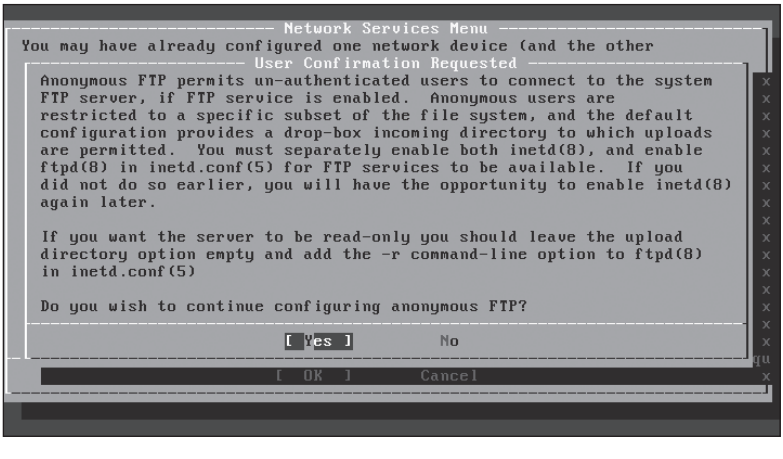

Рис. 25.4. Нажмите Yes

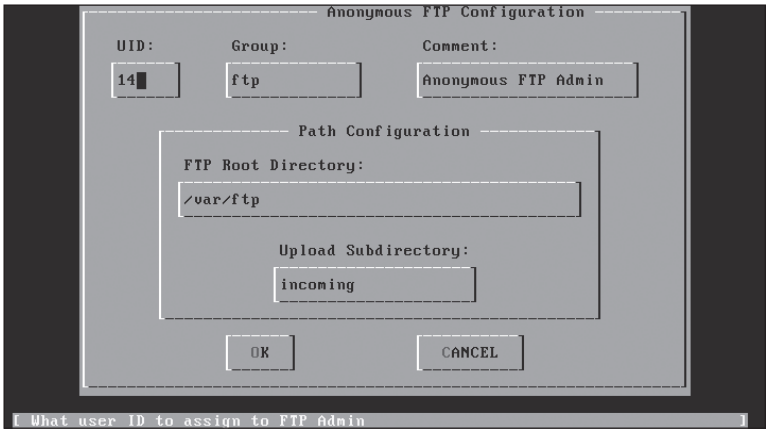

Рис. 25.5. Настройки анонимного FTP

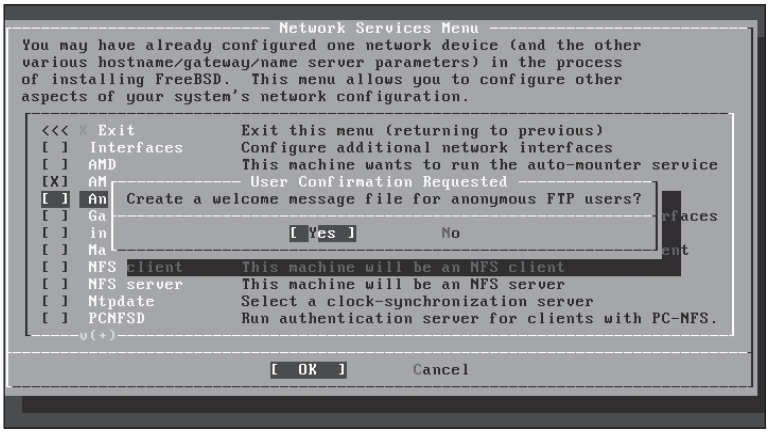

Рис. 25.6. Создать файл приветствия для анонимных пользователей?

Все, анонимный доступ настроен. Но с помощью конфигуратора sysinstall нельзя отключить анонимный доступ к серверу после того, как он включен. Лучший способ избавиться от анонимного доступа, если он вам не нужен или вы включили его по ошибке, — внести пользователя ftp в файл /etc/ftpusers.

Всем анонимным пользователям разрешается запись в каталог /var/ftp/incoming. Чтобы избавиться от этого, нужно в inetd.conf запустить программу `ftpd` с параметром `-r` (режим «только чтение»):

```
ftp stream tcp nowait root /usr/libexec/ftpd ftpd -r
```

Но тут есть одна неувязочка — тогда и обычные пользователи не смогут сохранять файлы в своих домашних каталогах, поэтому вместо параметра `-r` проще установить другие права доступа к каталогу /var/ftp/incoming:

```
# chmod 1555 /var/ftp/incoming
```

25.4. Настройка ProFTPD

Базовый сервер ftpd хорош для выполнения базовых задач. Да, он обеспечивает основные функции FTP-сервера. Но ничего более. Получившийся вариант уж больно классический. В реальных условиях может возникнуть множество реальных задач, например ограничить количество одновременно работающих пользователей или количество пользователей с одного IP-адреса. Сервер ftpd просто не сможет справиться с поставленной задачей, так как таких функций в нем не предусматривается.

Совсем другое дело ProFTPD. Как подчеркивает название сервера, данный сервер является «профессиональной» версией ftpd. Сервер обладает гибким конфигурационным файлом, позволяющим настроить FTP-сервер так, как вам нужно. Именно поэтому он заменил классический сервер wu-ftpd, использовавшийся по умолчанию в Linux много лет.

Установим ProFTPD из портов:

```
# cd /usr/ports/ftp/proftpd
# make install clean
```

Можно, конечно, воспользоваться утилитой pkg_add — это как вам будет удобнее. В Linux можно воспользоваться вашим стандартным менеджером пакетов, например:

```
sudo apt-get install proftpd
```

Для автоматического запуска сервера во FreeBSD нужно в /etc/rc.conf добавить строку:

```
proftpd_enable="YES"
```

Для управления сервером во FreeBSD используется команда:

```
# /usr/local/etc/rc.d/proftpd <start|restart|stop>
```

или

```
# /usr/local/etc/rc.d/proftpd.sh start
```

В Linux можно использовать команду service:

```
# service proftpd <start|restart|stop>
```

После установки сервера нужно отредактировать его конфигурационный файл: /usr/local/etc/proftpd.conf во FreeBSD или /etc/proftpd.conf в Linux. Обычно настройки по умолчанию подходят всем, но некоторые нужно изменить.

Повысить производительность сервера поможет отключение директивы IdentLookups, которая преобразует IP-адреса клиентов в доменные имена, а на это нужно время, поэтому если ее выключить, то наш сервер будет работать быстрее:

```
IdentLookups off
```

Директивы ServerName и ServerType задают имя и тип сервера. Имя сервера — это обычная строка, в которой вы можете написать все что угодно. Тип сервера

в нашем случае всегда будет standalone — автономный. Ведь запускать ProFTPD через inetd не имеет смысла. Использование директив:

```
ServerName "My FTP Server"
ServerType standalone
```

Обязательно установите директиву DefaultRoot в ~. В этом случае корневым для пользователя станет его домашний каталог и пользователь не получит доступ к файлам и каталогам, находящимся за пределами своего домашнего каталога (следовательно, пользователь не сможет прочитать другие конфигурационные файлы сервера):

```
DefaultRoot ~
```

По умолчанию для FTP-сервера используется порт 21, но вы можете установить любой другой порт с помощью директивы Port:

```
Port 21
```

Директива MaxInstances ограничивает количество дочерних процессов сервера. Если сервер используется в небольшой сети, количество дочерних процессов можно уменьшить до 10 (или даже до 5) — для экономии системных ресурсов:

```
MaxInstances 10
```

Ограничить число одновременно работающих пользователей можно с помощью директивы MaxClients. Сколько пользователей зарегистрировано на FTP-сервере? Теоретически все они могут работать одновременно. Но хватит ли системных ресурсов? Если сервер мощный, то максимальное число клиентов можно установить даже с небольшим запасом, например (чтобы не пришлось увеличивать это значение, когда число пользователей увеличится):

```
MaxClients 200
```

А если ресурсов, наоборот, мало, нужно установить низкое значение — кто не успел, тот опоздал:

```
MaxClients 20
```

Если сервер работает только в корпоративной сети, целесообразно ограничить количество пользователей с одного IP-адреса. Больше и быть не может:

```
MaxClientsPerHost 1
```

А вот если сервер будет использоваться в Интернете, тогда с одного IP-адреса может быть много клиентов. Предположим, что у вас сервер с музыкой, а у какого-то провайдера только один реальный IP-адрес и с помощью NAT все его пользователи используют этот адрес для обращения к узлам Интернета. В этом случае клиентов с одного адреса будет много:

```
MaxClientsPerHost 15
```

Количество соединений от одного пользователя ограничивается директивой MaxClientsPerUser. Для экономии системных ресурсов установим всего два соединения от одного пользователя (в локальной сети можно установить всего одно):

```
MaxClientsPerUser 2
```

Почему два соединения, а не одно? Представим, что пользователь обращается к вашему серверу по модемному соединению. Обрыв соединения. Но поскольку FTP-сессия еще не закрыта, пользователь после восстановления соединения с Интернетом не сможет сразу зайти на FTP-сервер, пока его сессия не будет закрыта автоматически. Ему придется подождать около 3–5 минут. А если допускается два соединения, то пользователь будет просто использовать второе соединение. В локальной сети обрывов нет, поэтому можно установить одно соединение от одного пользователя:

```
MaxClientsPerUser 1
```

Максимальное количество попыток входа целесообразно ограничить тремя, чтобы исключить возможность перебора пароля:

```
MaxLoginAttempts 3
```

Имя пользователя и группы, от имени которых запускается ProFTP, задаются директивами User и Group:

```
User  ftp
Group ftp
```

Чтобы разрешить анонимный доступ к серверу, нужно раскомментировать секцию Anonymous:

```
<Anonymous ~ftp>
User ftp
Group nogroup
# Пользователь anonymous – псевдоним для ftp
UserAlias anonymous ftp
# Все файлы принадлежат пользователю ftp
DirFakeUser on ftp
DirFakeGroup on ftp
#Максимальное количество анонимных клиентов
MaxClients 10
# Содержимое файла welcome.msg будет отображаться при входе пользователя
# Содержимое файла .message будет отображаться при первом изменении каталога
DisplayLogin welcome.msg
DisplayFirstChdir .message

# Запрещаем WRITE (запись) за пределами анонимного chroot
<Directory *>
<Limit WRITE>
DenyAll
</Limit>
</Directory>

# Запись файлов разрешается только в каталог /home/ftp/incoming.
<Directory incoming>
Umask 022 022
<Limit READ WRITE>
DenyAll
```

```
</Limit>
<Limit STOR>
AllowAll
</Limit>
</Directory>
</Anonymous>
```

Полный листинг конфигурационного файла приводить не стану, так как он довольно громоздкий за счет множества комментариев. Обладая минимальными знаниями английского языка (или умением использовать сервис translate.ru), вы без проблем разберетесь с ним сами, а все, на что нужно обратить ваше внимание, уже было изложено в этой главе.

Настройка веб-сервера Apache

26

26.1. Информация об Apache

Apache — это веб-сервер, входящий в состав практически всех свободных UNIX-систем (в коммерческих системах могут быть свои разработки). Apache точно есть в репозиториях всех дистрибутивов Linux, а также FreeBSD. Есть даже Windows- и MacOS-версии этого веб-сервера.

Не подумайте, что Apache — это единственный веб-сервер для UNIX. Кроме Apache есть и другие серверы: kHTTPd, SUN One Web Server, NCSA и др. Но Apache — самый популярный из них. Этого у него не отнимешь.

История Apache уходит корнями в проект NCSA. Сейчас Apache поддерживается организацией Apache Group, а в далеком 1994 году он поддерживался всего лишь одним разработчиком, который собрал вместе все дополнения и исправления к коммерческому NCSA и ушел из проекта NCSA. На базе всех этих дополнений и был разработан новый сервер — Apache. Первая версия Apache появилась в 1995 году, позже сервер стал полностью отдельной и независимой разработкой.

Почему же этот сервер стал настолько популярным? Основные причины — надежность и огромная гибкость настройки, которая достигается благодаря различным внешним модулям. Желаете поддержку PHP? Установите модуль mod_php — больше ничего делать не нужно, не нужно изменять конфигурационный файл, долго разбираться, почему та или иная функция не работает. Максимум, что от вас потребуется — это перезапуск Apache. Его конфигурационный файл довольно большой, но зато обилие разных директив конфигурации позволяет настроить сервер так, как вам нужно. Недостатков у него нет. Вообще нет. Фанаты всего компактного скажут, что он неповоротлив. Да, если сравнивать его с kHTTPd при единичных запросах. Но kHTTPd вряд ли выдержит ту нагрузку, с которой справляется Apache, да и вряд ли вы сможете настроить kHTTPd так гибко, как Apache. Зато kHTTPd идеально подойдет для использования во встроенных устройствах, например в каком-нибудь маршрутизаторе для запуска веб-интерфейса настройки.

26.2. Автоматический запуск Apache

Веб-сервер Apache устанавливается по умолчанию при установке FreeBSD (при условии, что вы выбрали все пакеты). Если сервер каким-то чудом не был установлен, тогда установите его из порта /usr/ports/www/apache22.

В Linux все зависит от дистрибутива. В одних дистрибутивах Apache установлен по умолчанию, в других нужно установить его из репозитория (см. главу 11). Имя пакета тоже зависит от дистрибутива и версии Apache. Обычно это apache2 или apach22. Проще всего выполнить поиск по строке «apache» и определиться, какой пакет нужно установить. Например, в случае с yum это делается так:

```
# yum search apache
```

Во FreeBSD сервер Apache нужно настроить на автоматический запуск, а вот в Linux этого делать не нужно — вся необходимая настройка будет выполнена автоматически при установке пакета apache22.

Для управления Apache во FreeBSD используются следующие команды:

```
# /usr/local/etc/rc.d/apache22 start
# /usr/local/etc/rc.d/apache22 stop
```

Можно также использовать скрипт apachectl (это как кому удобнее):

```
# /usr/local/sbin/apachectl start
# /usr/local/sbin/apachectl restart
# /usr/local/sbin/apachectl stop
```

Думаю, эти команды не нуждаются в объяснении. Первая запускает сервер, вторая — перезапускает его (например, после установки модуля или редактирования конфигурационного файла), а третья — останавливает.

В Linux используется команда service, но имя самого сервиса может изменяться в зависимости от дистрибутива и версии самого Apache. Мне встречались следующие имена сервиса: httpd, apache, apache22. Узнать, как точно называется сервис в вашем дистрибутиве, можно, просмотрев содержимое каталога /etc/init.d. В нем нужно найти службу, похожую на название сервера. Возможные варианты я вам указал ранее. Вот пример использования команды service:

```
# service apache start
# service apache restart
# service apache stop
```

А теперь разберемся, как организовать автоматический запуск сервера. Для этого откройте ваш /etc/rc.conf и добавьте в него строку:

```
apache22_enable=YES
```

Вот и все. Настало время настроить сам сервер. Но перед настройкой веб-сервера нужно убедиться, что DNS-сервер работает корректно. Если вы пока не настраивали DNS-сервер, а DNS-сервер провайдера недоступен (например, когда вы хотите запустить веб-сервер на машине, не подключенной к Интернету — для тестирования своих скриптов), то добавьте имя своего компьютера в файл /etc/hosts, например:

```
10.0.0.1    server        server.firma.ru
```

Запустите сервер и попробуйте подключиться к нему, используя браузер, например:

```
lynx http://10.0.0.1
```

Вы должны увидеть сообщение «It works!», что свидетельствует о нормальном запуске сервера (рис. 26.1).

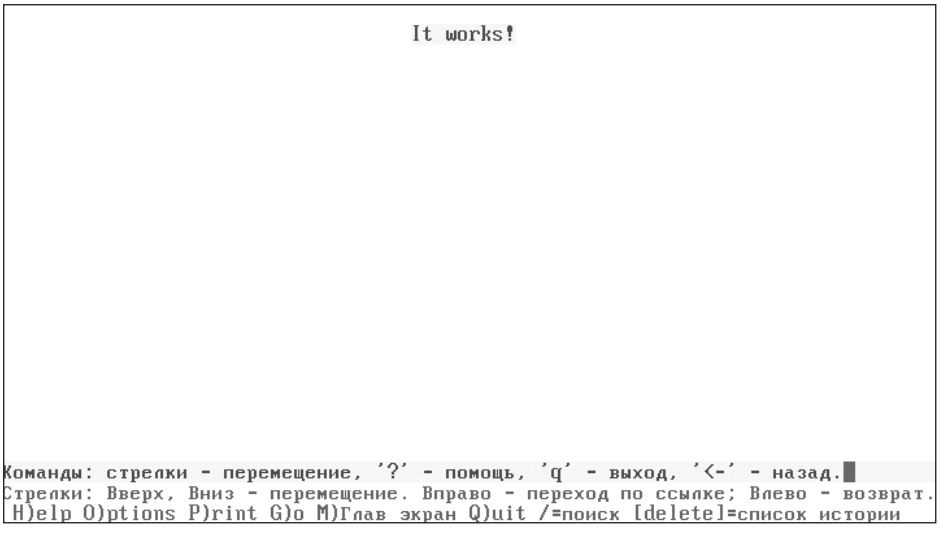

Рис. 26.1. Текстовый браузер lynx

26.3. Директивы файла конфигурации Apache

В отличие от некоторых других сетевых сервисов, у Apache несколько конфигурационных файлов. Все они находятся в каталоге /usr/local/etc/apache22 во FreeBSD и в каталоге /etc/httpd (или /etc/apache) в Linux. Хорошо хоть конфигурационный файл называется везде одинаково — httpd.conf. Кроме этого файла в каталоге Includes вы найдете дополнительные конфигурационные файлы; они выносятся в отдельный каталог для уменьшения основного файла httpd.conf. В каталоге extra находятся файлы настройки, относящиеся к виртуальным хостам, SSL и прочим дополнительным параметрам. Файлы mime.types и magic вы будете редактировать очень редко, если вообще будете. В первом содержится описание MIME-типов (неужели вам нужно будет добавить свой MIME-тип?), а во втором — настройки модуля mod_mime_magic.

Обычно вам придется редактировать только httpd.conf. Это довольно большой файл, даже с учетом того, что некоторые настройки вынесены в каталоги Includes и extra. Каждая директива конфигурационного файла тщательно прокомментирована, поэтому при его чтении вам даже не понадобится страница руководства.

Но как всегда есть один нюанс: все комментарии на английском языке. Кто-то его знает, а кто-то — нет. В этой главе вы найдете файл конфигурации с комментариями на русском языке — он поможет вам разобраться во всевозможных настройках сервера.

Директивы конфигурации сервера можно разделить на группы:

- общие — влияют на работу всего сервера. Сюда можно отнести директивы ServerName (задает имя сервера), ServerType (тип запуска сервера), Port (порт, на котором работает сервер), ServerRoot (корневой каталог сервера), DocumentRoot (каталог документов сервера) и т. д. Общие директивы описаны в табл. 26.1;
- журналирования — определяют имена файлов и расположение журналов сервера: ErrorLog (журнал ошибок), TransferLog (журнал передачи) и т. д.;
- ограничения доступа — позволяют ограничить доступ к разным объектам сервера, например вы можете закрыть доступ к какому-нибудь каталогу для всех пользователей, кроме пользователей локальной сети. Это директивы: Limit, Options, AllowOverride;
- производительности — влияют на производительность сервера, к этой группе относят директивы StartServers, MaxSpareServers, MinSpareServer и т. д.;
- постоянного соединения — обеспечивают поддержку постоянного соединения (Timeout, KeepAlive, KeepAliveTimeout);
- отображения каталога — управляют отображением каталога, позволяют задать особые параметры отображения каталогов сервера (Directorylndex, FancyIndexing и AddlconByType);
- обработки MIME-типов — настраивают сервер для обработки MIME-типов (DefaultType, AddEncoding, AddType, AddHandler и Action);
- виртуальных узлов — позволяют создать несколько виртуальных веб-серверов на одном физическом сервере (VirtualHost, BindAddress, Listen);
- обработки ошибок — определяют реакцию сервера на различные ошибки (к этой группе относится всего одна директива — ErrorDocument);
- перенаправления — позволяют перенаправить пользователя с одной страницы сервера на другую или даже на другой сервер (Redirect, Alias, ScriptAlias);
- все остальные директивы, не вошедшие в описанные ранее группы.

Таблица 26.1. Общие директивы Apache

Директива	Описание
ServerName	Задает имя сервера. Минимальная настройка Apache сводится к изменению этой директивы. Имя сервера должно быть прописано в DNS или хотя бы в файле /etc/hosts. Если укажете имя, которое Apache не сможет разрешить (которого нет в DNS или hosts), Apache не будет запущен
ServerRoot	Определяет корневой каталог сервера; не нужно изменять эту директиву! Ее изменение может понадобиться, только если у вас появилось огромное желание перенести корневой каталог сервера
ServerAdmin	Адрес электронной почты администратора

Директива	Описание
DocumentRoot	Корневой каталог документов. Вы хотите изменить главную страничку сервера? Тогда файл index.html (или index.php) нужно поместить в каталог, указанный в DocumentRoot
Listen	Адрес интерфейса, на котором будет работать Apache. Можно вообще не указывать эту директиву (можно закомментировать ее), тогда Apache будет доступен на всех интерфейса. Что это означает? Пусть у вас есть два интерфейса — один локальный (к вашей локальной сети), другой — для доступа к Интернету. Если директива Listen не указана, тогда к серверу смогут обратиться не только локальные пользователи, но и пользователи Интернета, если, конечно, межсетевой экран не запретил интернет-пользователям доступ к внутренним ресурсам. Тут все зависит от того, какой сервер вы настраиваете. Возможно, вам нужно настроить корпоративный веб-сервер, доступный только пользователям локальной сети
Port	Раньше использовалась для определения номера порта веб-сервера. Сейчас она не используется, а номер порта можно задать директивой Listen

Скажем так, если вам нужно «поднять» сервер еще вчера, тогда вы можете ограничиться только общими директивами. Больше ничего редактировать не нужно — сервер и так будет работать. Но рано или поздно возникнет необходимость более тонко настроить свой сервер.

26.4. Ограничение доступа к серверу

Для ограничения доступа к серверу можно использовать директивы Limit, AllowOverride, Options в блоке Directory. Блок Directory описывает параметры каталога, доступного по HTTP.

Начнем с первой директивы. Limit используется для ограничения доступа к каталогу. Рассмотрим пример использования Limit:

```
<Directory /corporate>
    <Limit GET POST>
        order deny,allow
        deny form all
        allow from corp.ru
    </Limit>
</Directory>
```

Limit позволяет указать, для каких методов передачи данных будет задействовано ограничения. Мы выбрали оба метода — GET и POST. Затем мы задаем порядок ограничение — сначала запретить, потом разрешить:

```
order deny,allow
```

Потом мы запрещаем доступ к каталогу /corporate всем пользователям:

```
deny form all
```

А разрешаем только пользователям из домена corp.ru:

```
allow from corp.ru
```

Доступ к каталогу /corporate должны получить только компьютеры нашей сети (домен corp.ru). Вместо имени домена можно указать адрес сети, например:

```
allow from 192.168.1.
```

Теперь перейдем к директиве AllowOverride. Каждый раз редактировать файл конфигурации Apache и перезагружать сервер (имеется в виду командой service, а не командой reboot), когда нужно определить параметры доступа к каталогу, не очень удобно. Сервер Apache поддерживает файлы доступа .htaccess. Такой файл можно поместить в каталог, доступ к которому вы хотите ограничить, а сами правила доступа можно прописать в этом файле. Преимущество заключается в том, что вам не нужно перезагружать сервер после изменения файла .htaccess. Вот пример .htaccess-файла, запрещающего доступ к каталогу, в котором он расположен, всем узлам, кроме localhost:

```
<limit GET, POST>
order deny,allow
deny from all
allow from localhost
</limit>
```

Директива AllowOverride определяет «полномочия» файлов .htaccess и может принимать следующие значения:

- None — игнорирование файлов .htaccess. Это позволяет повысить производительность сервера (поскольку не нужно обращать внимание на .htaccess-файлы и обрабатывать их), но создаст определенные неудобства, особенно если вы используете сервер не в гордом одиночестве. Ведь другие пользователи не смогут изменить параметры каталогов. Тогда они начнут вас просить сделать это с помощью глобального конфигурационного файла Apache, что, в свою очередь, создаст дискомфорт для вас, как для администратора.
- All — разрешает .htaccess-файлы, в которых можно использовать практически все глобальные параметры сервера; из соображений безопасности забудьте об этом режиме.
- Options — разрешает использовать в .htaccess-файлах директиву Options.
- Limit — разрешает использовать в .htaccess-файлах директиву Limit.
- FileInfo — допускается использование в .htaccess-файлах директивы AddType и AddEncoding.
- AuthConfig — допускается использование в .htaccess-файлах директивы аутентификации (AuthName, AuthType и др.).

Если вы не запретили использование файлов .htaccess, рекомендуется добавить секцию Files в файл httpd.conf, защищающую файлы .htaccess от просмотра посредством HTTP (изменить и просмотреть .htaccess-файлы можно по FTP):

```
<Files .htaccess>
   deny from all
</Files>
```

Директива Options позволяет ограничить различные возможности сервера, она может принимать такие значения:

- None — все функции запрещены.
- All — все функции разрешены.
- Includes — допускается использование SSI (Server-Side Includes), но помните, что SSI снижает производительность сервера.

- IncludesNoExec — частично разрешает SSI: запрещен только запуск внешних программ из SSI.
- Indexes — разрешает или запрещает просмотр содержимого каталога, в котором отсутствует индексный файл (см. ниже).
- ExecCGI — разрешено использование CGI-скриптов.
- FollowSymLinks — позволяет использовать символические ссылки, указывающие на любой файл, не используйте эту опцию — она не безопасна!
- SymLinksIfOwnerMatch — похожа на FollowSymLinks, но более безопасна, потому что разрешает ссылку, если она указывает на файл, принадлежащий тому же пользователю, что и ссылка.

26.5. Управление протоколированием

Обычно вам не придется изменять значения директив ErrorLog (задает имя протокола ошибок) и TransferLog (имя протокола передач). Но, возможно, вам захочется изменить значение HostNameLookup. Когда пользователь запрашивает с вашего сервера файл, в журнал передач заносится время передачи файла, имя самого файла и IP-адрес клиента. Если директива HostNameLookup включена (значение on), то вместо IP-адреса будет занесено имя узла. Но разрешение IP-адреса в доменное имя требует дополнительного времени, что повышает нагрузку на сервер и увеличивает трафик. Подумайте дважды, прежде чем включить HostNameLookup.

26.6. Директива Redirect

Бывает так, что вы переименовали какой-то каталог или вообще перенесли его на другой сервер. Но пользователи все еще обращаются по старому адресу. Чтобы пользователи могли получить доступ к данным, нужно использовать перенаправление. Следующий пример перенаправит запросы к каталогу /dir на каталог /dir2:

```
Redirect /dir /dir2
```

Можно перенаправить запросы даже на другой сервер, например:

```
Redirect /dir www.piter.com
```

26.7. Управление отображением каталога. MIME-типы

Представим, что мы вводим адрес http://www.server.ru/ в строку браузера. Браузер отправляет запрос серверу. Но какой файл передаст пользователю сервер? Ведь мы указали протокол (http://), имя сервера (www.server.ru), но не указали имя файла, который сервер должен передать нам. Какой файл будет передан, зависит от значения директивы DirectoryIndex. Вот ее примерное значение:

```
DirectoryIndex index.html index.php index.htm
```

Сервер сначала будет искать в корневом каталоге документов (ведь другой каталог в URL не задан) файл index.html. Если он существует, сервер передаст его браузеру пользователя. Если такого файла нет, будет передан файл index.php. Но поскольку наш сервер оснащен (точнее, будет оснащен) модулем mod_php, то пользователь получит не PHP-код, содержащийся в файле index.php, а результат выполнения этого кода интерпретатором PHP. Если и файла index.php не будет, тогда будет передан файл index.htm.

Наконец, что будет возвращено пользователю, если ни один из файлов, указанных в DirectoryIndex, не найден? Тогда сервер отобразит содержимое каталога. Вам это не нужно? Вы не хотите, чтобы сервер возвращал содержимое каталога, если в нем отсутствует индексный файл, заданный директивой DirectoryIndex? Тогда используйте опцию -Indexes:

```
<Directory />
…
Options -Indexes
…
</Directory>
```

Хотя намного безопаснее указать так:

```
Options None
```

Директива FancyIndexing определяет, как будет отображено содержимое каталога, если вы разрешили его вывод. Если FancyIndexing включена (значение on), тогда при выводе оглавления будут использоваться значки папки, а значки файла будут соответствовать MIME-типу файла, что делает вывод каталога более привлекательным:

```
FancyIndexing on
```

Добавить оригинальности вашему серверу может и улучшенная поддержка MIME-типов. Например, вы можете добавить обработку собственного формата файлов. Добавляется поддержка нового типа так:

```
AddType text/myml
AddHandler text/myml myml
Action text/myml /cgi-bin/myml-parse
```

Мы добавили поддержку типа myml (сокращение от My Markup Language). Базовым для нового типа есть тип text (текстовый файл). Можно использовать и другую базу, например image (изображение). Для обработки типа myml будет выполнена программа /cgi-bin/myml-parse.

Хотя правильнее добавлять новый тип в файле mime.types, а в файле httpd.conf корректнее прописывать только обработчик (Action) этого типа. Вот фрагмент файла mime.types, в который я добавил новый тип:

```
# MIME type                    Extensions
text/myml                      ml
application/activemessage
application/andrew-inset       ez
application/applefile
...
```

26.8. Обработка ошибок

У некоторых администраторов есть огромное желание сделать свой сервер непохожим на все остальные. Чего только они не придумают. Один из способов выделить свой сервер из серой массы — это нестандартные сообщения об ошибках. Поскольку ошибка 404 (файл отсутствует) возникает чаще всего, то чаще всего и изменяют сообщение об этой ошибке. Делается это так:

```
ErrorDocument 404 /404.html
```

В случае возникновения ошибки 404 сервер передаст браузеру файл 404.html, находящийся в корневом каталоге документов сервера. Что поместить в этот файл — зависит от вашей фантазии.

Теоретически можно задать собственный файл для любой HTTP-ошибки, но обычно в этом нет необходимости, потому что другие ошибки возникают редко.

26.9. Поддержка PHP

Сейчас мы добавим поддержку PHP в наш Apache. Вообще современный веб-сервер сложно представить без PHP[1]. Даже не представляю, почему до сих пор Apache не поддерживает PHP по умолчанию? А разработчикам дистрибутивов Linux можно тоже сделать укор — неужели так сложно добавить еще одну зависимость для пакета apache от пакета mod_php (или php5 — все зависит от названия пакета)? Тогда бы при установке Apache автоматически устанавливалась модель поддержки PHP.

В Linux для поддержки PHP нашим веб-сервером нужно установить пакет php5. Заодно установите пакет php5-mysql — он добавит поддержку MySQL в интерпретатор PHP. И конечно же, нужно установить сам MySQL: mysql-server-5.0 и mysql-client-5.0 (клиентская часть, которая вам обязательно понадобится в процессе работы). Команда установки зависит от вашего дистрибутива, например в Debian команда будет выглядеть так:

```
sudo apt-get install php5 php5-gd php5-mysql mysql-server-5.0 mysql-client-5.0
```

Пакет php5-gd — это библиотека для работы с графическими файлами, пригодится для создания счетчиков на PHP и фотогалерей. Необходима для многих серьезных CMS.

ПРИМЕЧАНИЕ

В некоторых дистрибутивах для поддержки PHP недостаточно установить пакет php5. Нужно установить пакет mod_php5 или libapache2-mod-php5.

[1] Можно, очень даже можно. Можно использовать perl, можно вообще использовать python. Все возможные способы «веб-программирования» даже не перечислить (не говоря уже о программировании под принципиально иные серверы). И поскольку нельзя заранее предсказать, что именно будет использоваться в том или ином случае, вполне корректна позиция разработчиков Apache и создателей дистрибутивов — пусть каждый выбирает для себя то, что нужно именно ему и его серверу. Включая и СУБД — например, PostgreSQL вместо MySQL.

Во FreeBSD для поддержки PHP и MySQL введите команды:

```
# cd /usr/ports/lang/php5
# make install clean

# cd /usr/ports/lang/php5-extensions
# make install clean

# cd /usr/ports/databases/mysql50-server
# make install clean
# mysql_install_db --user=mysql
# chown -R mysql /var/db/mysql/
# chgrp -R mysql /var/db/mysql/
# /usr/local/bin/mysqld_safe -user=mysql &
```

Первые две команды устанавливают сам PHP5, вторые две команды устанавливают расширения PHP. В процессе установки нужно выбрать расширения, которые вы желаете установить. Нужно как минимум два расширения — GD (графическая библиотека) и MYSQL (поддержка MySQL).

Далее идут команды установки MySQL-сервера. При установке сервера MySQL обязательно убедитесь, что опция WITH_OPENSLL включена. Команда mysql_install_db устанавливает начальную базу данных MySQL, затем следуют две команды установки прав доступа для корректной работы MySQL. Последняя команда запускает MySQL-сервер. Для автоматического запуска сервера MySQL нужно добавить в файл /etc/rc.conf следующую строку:

```
mysql_enable="YES"
```

Напомню, что в Linux MySQL-сервер автоматически настраивается на запуск при загрузке системы. Единственное, что нужно сделать, — это запустить его вручную после установки пакетов:

```
# service mysql start
```

Практически все. Осталось только в корневом каталоге документов (задан директивой DocumentRoot) создать файл test.php вот с таким содержимым:

```
<?
phpinfo();
?>
```

Перезапустите сервер Apache:

```
# /usr/local/etc/rc.d/apache22 restart (FreeBSD)
# service apache restart
```

Введите команду для просмотра test.php:

```
lynx http://127.0.0.1/test.php
```

Если вы увидели тестовую страницу PHP, вы все сделали правильно (рисунок 26.2). А вот если вы увидели код файла test.php, тогда вы что-то сделали неправильно. Одна из самых вероятных причин — вы просто забыли перезапустить Apache.

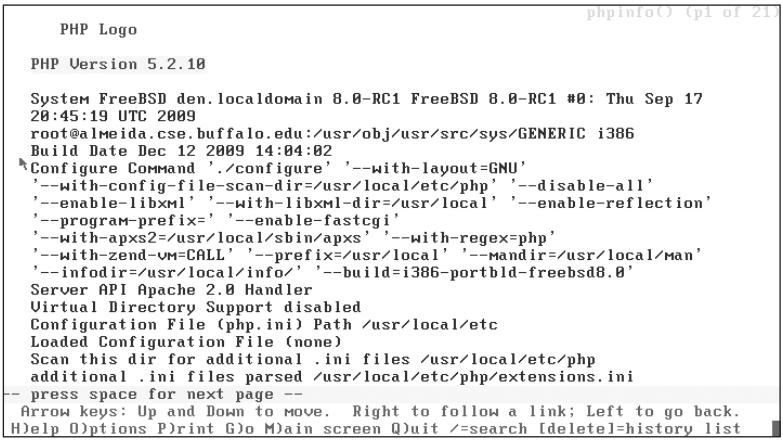

Рис. 26.2. Тестовая страница PHP

26.10. Оптимизация сервера

Представим обычную сессию работы пользователя с вашим сайтом. Пользователь заходит на стартовую страницу сервера, браузер отправляет запрос — что-то вроде этого:

```
GET /
```

Сервер возвращает главную страницу (файл index.html) и закрывает соединение. Затем пользователь переходит по ссылке на страницу контактов. Браузер отправляет запрос:

```
GET /contacts.html
```

Сервер отправляет пользователю содержимое файла contacts.html и закрывает соединение. Затем пользователь переходит на третью страницу — на страницу price.html:

```
GET /price.html
```

Сервер передает файл price.html и снова закрывает соединение. Как можно оптимизировать работу сервера? Правильно, не закрывать соединение. Ведь сервер может немного и подождать, пока клиент запросит следующую страницу. Мы сэкономим время на двух лишних установках соединений. В конечном итоге сервер будет работать быстрее, и пользователи будут вам благодарны.

Вы можете разрешить клиенту (то есть браузеру пользователя) отправлять несколько запросов за один сеанс. Для этого установите директивы файла конфигурации httpd.conf так:

```
KeepAlive On
MaxKeepAliveRequests 100
KeepAliveTimeout 15
```

Первая директива разрешает постоянные соединения, то есть теперь клиент может за один сеанс отправлять несколько запросов. Вторая директива определяет максимальное число запросов за один сеанс. А последняя директива задает тайм-аут постоянного соединения в секундах. Если пользователь за 15 секунд не отправил следующий запрос, сервер закроет соединение.

Повысить производительность могут также следующие директивы:

```
StartServers 5
MinSpareServers 5
MaxSpareServers 10
MaxClients 50
MaxRequestsPerChild 0
```

Первый параметр определяет число процессов сервера, запускаемое при старте самого сервера. То есть при запуске сервера будет работать 5 копий сервера. Каждая копия может принимать запросы от пользователей. Не перестарайтесь! С одной стороны, чем больше копий сервера, тем больше запросов можно обработать за одну единицу времени. С другой стороны, много копий Apache могут забить оперативную память. Также нужно помнить, что при использовании постоянных соединений число запускаемых серверов нужно увеличить, поскольку каждая копия сервера будет обслуживать одного клиента как минимум KeepAliveTimeout секунд. Помните об этом.

Директива MinSpareServers — минимальное число копий сервера, которые будут ожидать запросы клиентов. А MaxSpareServers — максимальное число копий. У нас 5 копий запускается при запуске системы, а 10 может быть максимально запущено. Максимальное число клиентов, одновременно обращающихся к серверу, задается директивой MaxClients.

Максимальное число запросов для каждого дочернего процесса задается последней директивой. Если она равна 0, то количество запросов не ограничивается.

26.11. Конфигурационный файл httpd.conf

Как я вам и обещал, мы рассмотрим файл `httpd.conf` с моими комментариями на русском языке (листинг 26.1). Некоторые стандартные комментарии я сократил, чтобы сделать листинг немного меньше.

Листинг 26.1. Файл httpd.conf

```
#
# Основной конфигурационный файл HTTP-сервера Apache.
# Подробную информацию можно получить по адресу:
# <URL:http://httpd.apache.org/docs/2.2>
#
# Соглашение об именах конфигурационных файлов
# Если имя начинается со слэша (/), значит, указан
# абсолютный путь к файлу. В противном случае к имени
# файла будет добавлено значение директивы ServerRoot
```

```
#
# ServerRoot: основной каталог с конфигурационными
# файлами, журналами ошибок и событий. Также в нем
# вы найдете файлы блокировок и PID-файлы
#
# Значение по умолчанию - /usr/local, не нужно его изменять!

ServerRoot "/usr/local"

#
# Listen: задает IP-адрес и/или порт интерфейса, который
# будет прослушиваться сервером в ожидании запросов клиентов
# Примеры значений (адрес и порт; только порт):
#
#Listen 192.168.56.78:80
#Listen 80

#
# Поддержка динамически разделяемых объектов (модулей)
# Dynamic Shared Object (DSO) Support
#
# Если вам нужно включить определенный модуль, нужно
# раскомментировать соответствующую ему строку.
#
# Весь список довольно длинный, поэтому я его сократил.
#
# Список модулей, встроенных статически (при компиляции
# Apache), можно просмотреть командой httpd -l
#
#
LoadModule authn_file_module libexec/apache22/mod_authn_file.so
LoadModule authn_dbm_module libexec/apache22/mod_authn_dbm.so
LoadModule authn_anon_module libexec/apache22/mod_authn_anon.so
...
LoadModule dnssd_module       libexec/apache22/mod_dnssd.so
LoadModule php5_module        libexec/apache22/libphp5.so

<IfModule !mpm_netware_module>
<IfModule !mpm_winnt_module>
#
# Имя пользователя и группы, от имени которых будет запускаться
# сервер Apache. Обычно не нужно изменять эти значения.

User www
Group www

</IfModule>
</IfModule>

# *** Параметры основного сервера ***
# ***********************************
#
# Все директивы в файле httpd.conf относятся к основному
# серверу. Параметры виртуальных серверов находятся в файле
# extra/httpd-vhosts.conf
#
```

продолжение ⇩

Листинг 26.1 *(продолжение)*

```
# Директивы, заданные в httpd.conf, влияют и на виртуальные
# серверы, если аналогичные параметры не заданы
# в файлах параметров виртуальных узлов.

# ServerAdmin: определяет электронный адрес администратора
#
ServerAdmin you@example.com

# ServerName: имя сервера
# Указанное здесь имя сервера должно быть "прописано"
# в системе DNS. Если вы пока не настроили DNS-сервер или же
# планируете использовать Apache для отладки собственных
# PHP-сценариев, тогда укажите имя, заданное в ServerName,
# в файле /etc/hosts.

ServerName myserver:80

#
# DocumentRoot: корневой каталог документов сервера.
# В этот каталог нужно поместить HTML/PHP-файлы
#
DocumentRoot "/usr/local/www/apache22/data"

# Директива Directory задает параметры каталога.
# Параметры корневого каталога по умолчанию:
#
<Directory />
    AllowOverride None
    Order deny,allow
    Deny from all
</Directory>

#
# Далее следуют параметры корневого каталога документов
# (см. DocumentRoot). Блок Directory для этого каталога
# я сократил (из соображения экономии бумаги) - вы всегда
# можете просмотреть его в своем конфигурационном файле,
# тем более что все директивы были подробно описаны
# ранее
#
...
#
# DirectoryIndex: определяет имена индексных файлов
#
<IfModule dir_module>
    DirectoryIndex index.html index.php index.cgi index.htm
</IfModule>

#
# Запрещаем доступ к файлам .htaccess
#
<FilesMatch "^\.ht">
    Order allow,deny
    Deny from all
```

```
    Satisfy All
</FilesMatch>

#
# ErrorLog: имя журнала ошибок
ErrorLog "/var/log/httpd-error.log"

#
# LogLevel: уровень журналирования (для журнала ошибок)
# Может принимать значения:
# debug – протоколировать все сообщения сервера
# info – протоколировать информационные и более
# "серьезные" сообщения – предупреждения, ошибки и т. д.
# notice – протоколировать уведомления и более "серьезные"
# warn – протоколировать предупреждения и более "серьезные"
# error – протоколировать только ошибки
# crit - протоколировать только критические ошибки
# alert и emerg – протоколировать самые серьезные ситуации
#
LogLevel warn

<IfModule log_config_module>
    #
    # Формат журналов
    # для директивы CustomLog
    #
    LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
    LogFormat "%h %l %u %t \"%r\" %>s %b" common

   <IfModule logio_module>
   # Включите mod_logio.c для использования %I and %O
   LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\" %I %O" combinedio
   </IfModule>

# Позволяет записывать события в пользовательский,
# хотя правильнее бы сказать - администраторский,
# файл журнала. Формат файла определяется директивами, описанными
# выше

    CustomLog "/var/log/httpd-access.log" combined
</IfModule>

<IfModule alias_module>
    #
    # Redirect: перенаправление (директива была описана ранее)
    # Например:
    # Redirect permanent /den http://www.dkws.net

    #
    # Alias: создает псевдонимы файловой системы
    # Можно использовать для доступа к файлам,
    # которые находятся за пределами DocumentRoot
    # Пример:
    # Alias /webpath /full/filesystem/path
    #
```

продолжение ⇩

Листинг 26.1 *(продолжение)*

```
    #
    # ScriptAlias: расположение CGI-сценариев
    #
    ScriptAlias /cgi-bin/ "/usr/local/www/apache22/cgi-bin/"

</IfModule>

<IfModule cgid_module>
    #
    #
    #Scriptsock /var/run/cgisock
</IfModule>

# Параметры каталога для CGI-сценариев
<Directory "/usr/local/www/apache22/cgi-bin">
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all
</Directory>

#
# DefaultType: задает MIME-тип по умолчанию
# Сервер будет использовать этот тип для файлов, если
# ему не удастся определить их тип. Значение по умолчанию
# text/plain, но я рекомендую использовать тип
# "application/octet-stream", чтобы сервер не пытался
# передавать двоичные файлы как текст. А вот текстовые файлы
# в любом случае будут переданы корректно.
DefaultType application/octet-stream


<IfModule mime_module>
    #
    # TypesConfig имя файла, где описываются MIME-типы
    #
    TypesConfig etc/apache22/mime.types

    #
    # AddType добавляет описание нового MIME-типа
    #
    #AddType application/x-gzip .tgz
    #
    # С помощью AddEncoding можно распаковывать информацию на лету
    # Но эта функция поддерживается не всеми браузерами
    #
    #AddEncoding x-compress .Z
    #AddEncoding x-gzip .gz .tgz
    #
    #
    AddType application/x-compress .Z
    AddType application/x-gzip .gz .tgz

    #
```

```
    # AddHandler задает обработчик файлов определенного типа
    # Данная директива была описана выше

    #AddHandler cgi-script .cgi

    #AddHandler type-map var

   </IfModule>

# Модуль mod_mime_magic помогает серверу определить
# MIME-тип файла. Директива MIMEMagicFile задает
# имя файла с подсказками относительно MIME-типов

#
#MIMEMagicFile etc/apache22/magic

#
# Реакция на ошибки
# В качестве реакции на ошибку можно передать обычный текст,
# содержимое файла, использовать локальное перенаправление
# и даже перенаправление на внешний сервер. Примеры:
#ErrorDocument 500 "Сервер временно недоступен"
#ErrorDocument 404 /404.html
#ErrorDocument 404 "/cgi-bin/missing_handler.pl"
#ErrorDocument 402 http://www.dkws.org.ua/404
#

# *** Дополнительные параметры сервера ***
#
# Дополнительные параметры обычно выносят в конфигурационные
# файлы, находящиеся в каталоге etc/apache22/extra/, чтобы
# не забивать основной конфигурационный файл
# Далее идут директивы Include, подключающие файлы
# с дополнительными настройками

# Пул сервера: здесь прописываются директивы MaxSpareServers и др.
#Include etc/apache22/extra/httpd-mpm.conf

# Сообщения об ошибках на разных языках
#Include etc/apache22/extra/httpd-multilang-errordoc.conf

# Здесь указываются параметры, относящиеся к красивому
# выводу каталога (FancyIndexing)
#Include etc/apache22/extra/httpd-autoindex.conf

# Настройки разных языков
#Include etc/apache22/extra/httpd-languages.conf

# Пользовательские домашние каталоги: каждый пользователь
# может создать свою домашнюю страничку, страницы пользователей
# хранятся в их домашних каталогах, чтобы не предоставлять
# пользователям доступ к системному каталогу DocumentRoot
#Include etc/apache22/extra/httpd-userdir.conf

# Различная информация
#Include etc/apache22/extra/httpd-info.conf
```

продолжение ⇩

Листинг 26.1 *(продолжение)*

```
# Виртуальные узлы
#Include etc/apache22/extra/httpd-vhosts.conf

# Доступ к руководству Apache HTTP Server
#Include etc/apache22/extra/httpd-manual.conf

# Параметры WebDAV
#Include etc/apache22/extra/httpd-dav.conf

# Прочие параметры по умолчанию
#Include etc/apache22/extra/httpd-default.conf

# Безопасные соединения (SSL/TLS)
#Include etc/apache22/extra/httpd-ssl.conf
#
# Необходима поддержка модуля mod_ssl
#
<IfModule ssl_module>
SSLRandomSeed startup builtin
SSLRandomSeed connect builtin
</IfModule>

# Подключаем все остальные конфигурационные файлы из каталога
# Includes
Include etc/apache22/Includes/*.conf
```

Почтовый сервер 27

27.1. Необходимое программное обеспечение

Для настройки почтового сервера нам понадобится как минимум две программы — MTA (Mail Transfer Agent, агент передачи почты) и IMAP/POP3-сервер. Первая программа выполняет функцию сервера SMTP (Simple Mail Transfer Protocol, Простой протокол передачи почты), то есть отправляет почту. А вторая программа предоставляет доступ к почтовым ящикам пользователей нашей системы, то есть позволяет получить почту, адресованную нашим пользователям.

Но две программы — это самая простая конфигурация. Не для того вы покупали эту книгу, чтобы настроить простейший почтовый сервер. Нам понадобится еще две программы — это сервер баз данных MySQL для хранения настроек (в том числе и учетных записей пользователей) и программа (точнее библиотека) для SASL-аутентификации (чтобы кто угодно не смог отправить почту через наш сервер — эдакая минимальная защита от спама). Конечно, и это еще не предел. Можно еще добавить две программы — спам-фильтр и антивирус, но в этой книге подробно установка последних двух программ рассматриваться не будет. Позже я вам объясню почему. А пока определимся, какие программы мы будем использовать в качестве SMTP- и IMAP/POP3-сервера.

Классическим SMTP-сервером является программа sendmail. Но у нее есть ряд недостатков. Ранние «косяки» вроде дыр в безопасности уже давно устранены, однако огромный и запутанный файл конфигурации остался. Хочется выбрать SMTP-сервер с более прозрачным и понятным конфигурационным файлом. Так будет проще настраивать сервер — и сейчас, и в будущем. Я выбирал между Exim и Postfix: оба — отличные MTA, но для этой книги я выбрал Postfix, а Exim обязательно опишу в какой-нибудь другой книге.

Для IMAP/POP3-сервера будем использовать самый популярный сервер Courier-IMAP. Наверное, нет ни одного UNIX-администратора, который бы не настраивал Courier-IMAP. Вообще-то связка Postfix + Courier-IMAP не является чем-то экзотическим, вот если бы вместо Courier-IMAP я выбрал Dovecot или

иной, еще менее популярный сервер, вот тогда другое дело. А Postfix + Courier-IMAP — это почти классика для тех, кто хочет уйти от sendmail.

Можно все настраивать без MySQL, но конфигурация с MySQL лично мне нравится больше, чем без такового. Ведь в этом случае мы получаем возможность удобного и удаленного управления сервером. Я вам соврал, когда говорил, что мы будем использовать еще две дополнительных программы. Мы установим еще одну программу — PostfixAdmin — это веб-интерфейс для нашего почтового сервера. В итоге для управления сервером вы можете вообще к нему не подходить, а интерфейс PostfixAdmin наверняка удобнее, чем управление по SSH. Правда, для работы PostfixAdmin нам придется «поднять» веб-сервер Apache, но ведь мы это сделали в прошлой главе, так что никаких дополнительных неудобств.

В этой книге мы рассматриваем и Linux, и FreeBSD. Но, как мы уже договорились, серверные приложения будут рассматриваться с «уклоном» в сторону FreeBSD. Поэтому все сказанное здесь верно для FreeBSD. Описанную конфигурацию можно использовать и в Linux, но тогда вам придется самостоятельно установить пакеты и определить, как называются конфигурационные файлы — пути к ним будут другие, а вот имена файлов будут такими же.

27.2. Подготовительные работы

27.2.1. Установка MySQL

Перед установкой и настройкой Postfix и Courier-IMAP нужно установить сервер баз данных MySQL. Ему отводится очень важная роль в нашей конфигурации. Если при чтении предыдущей главы вы уже установили сервер MySQL, то ничего делать не нужно. А вот если вы этого не сделали, тогда у вас есть возможность установить самую последнюю версию MySQL (в прошлой главе мы установили версию 5.0, а сейчас установим версию 5.5):

```
# cd /usr/ports/databases/mysql55-server
# make install clean
# mysql_install_db
# chown mysql /var/db/mysql
# chgrp mysql /var/db/mysql
```

Первые две команды устанавливают порт mysql55-server, далее устанавливаются служебная база данных MySQL и права доступа для пользователя mysql, от имени которого будет запускаться наш сервер.

После этого запустим MySQL-сервер в безопасном режиме (первый запуск рекомендуется производить именно в нем, пока мы не установим пароль root):

```
# /usr/local/bin/mysqld_safe -user=mysql &
```

Теперь нужно установить пароль root пользователя MySQL-сервера. Не путайте этого пользователя с системным пользователем root, также желательно, чтобы их пароли отличались:

```
# mysqladmin -u root password ваш_пароль
```

Осталось только организовать автоматический запуск MySQL-сервера, добавьте следующую строчку в ваш /etc/rc.conf:

```
mysql_enable="YES"
```

27.2.2. Установка OpenSSL и Cyrus-sasl2

OpenSSL нужен для генерации ключей и сертификатов нашего почтовика, а библиотека Cyrus-sasl2 нужна для SASL-аутентификации на SMTP-сервере. Установим OpenSSL и Cyrus-sasl2:

```
# cd /usr/ports/security/openssl
# make install clean

# cd /usr/ports/security/cyrus-sasl2
# make install clean
```

При сборке Cyrus-sasl2 нужно включить опции MYSQL, LOGIN, PLAIN, CRAM, DIGEST. После сборки библиотеки Cyrus-sasl2 нужно откомпилировать библиотеку courier-authlib:

```
# cd /usr/ports/security/courier-authlib
# make install clean
```

При сборке этой библиотеки нужно включить только опцию WITH_MYSQL, остальные опции нам не нужны. После сборки библиотеки нужно добавить следующую строку в rc.conf:

```
courier_authdaemond_enable="YES"
```

Практически все. Осталось отредактировать конфигурационный файл /usr/local/etc/authlib/authdaemonrc — это конфигурация демона аутентификации. Отредактируйте его так, как показано в листинге 27.1.

Листинг 27.1. Файл /usr/local/etc/authlib/authdaemonrc

```
authmodulelist="authmysql"
authmodulelistorig="authmysql"
daemons=10
authdaemonvar=/var/run/authdaemond
subsystem=mail
DEBUG_LOGIN=0
DEFAULTOPTIONS="wbnodsn=1"
```

Также нужно отредактировать конфигурационный файл /usr/local/etc/authlib/authmysqlrc, отвечающий за MySQL-аутентификацию (листинг 27.2).

Листинг 27.2. Файл /usr/local/etc/authlib/authmysqlrc

```
MYSQL_SERVER            localhost
MYSQL_USERNAME          postfix
MYSQL_PASSWORD          пароль
MYSQL_PORT              0
MYSQL_OPT               0
MYSQL_DATABASE          postfix
MYSQL_USER_TABLE        passwd
MYSQL_CRYPT_PWFIELD     crypt
```

продолжение ⇨

Листинг 27.2 *(продолжение)*

```
MYSQL_UID_FIELD         3333
MYSQL_GID_FIELD         3333
MYSQL_LOGIN_FIELD       id
MYSQL_HOME_FIELD        home
MYSQL_NAME_FIELD        name
```

Можете оставить все параметры как есть, измените только пароль MySQL-пользователя. Остальные параметры изменять не нужно (разве что можно изменить название базы данных — тоже выделена жирным).

Автоматический запуск демона аутентификации мы уже обеспечили, осталось только запустить его:

```
# /usr/local/etc/rc.d/courier-authdaemond start
```

27.3. Установка и настройка IMAP/POP3-сервера

Вся подготовительная работа завершена, и мы можем установить POP3- и SMTP-сервер. Начнем с IMAP/POP3-сервера. Не будем изменять традиции и установим его из портов, к тому же при установке из портов мы можем включить необходимые нам опции:

```
# cd /usr/ports/mail/courier-imap/
# make install clean
```

Включите опции TRASHQUOTA и AUTH_MYSQL. Первая опция добавляет поддержку квот почтовых ящиков, а вторая — включает поддержку аутентификации с использованием MySQL.

После установки нужно отредактировать файл /usr/local/etc/courier-imap/pop3d. Файл не очень большой, но приводить его полностью не вижу смысла. Найдите следующие параметры и установите их так, как показано далее:

```
POP3AUTH="PLAIN LOGIN CRAM-MD5"
POP3AUTH_ORIG="PLAIN LOGIN CRAM-MD5
POP3AUTH_TLS="PLAIN LOGIN CRAM-MD5
POP3AUTH_TLS_ORIG="PLAIN LOGIN CRAM-MD5
POP3_PROXY=0
PORT=110
ADDRESS=0
```

Мы описываем поддерживаемые типы аутентификации. Можете изменить параметр ADDRESS — он задает IP-адрес интерфейса, на котором должен работать IMAP/POP3-сервер. Значение 0 означает, что должен работать на всех интерфейсах. Параметр POP3_PROXY задает IP-адрес POP3-прокси. Если такового нет в вашей сети, укажите 0.

Запустим IMAP/POP3-сервер:

```
# /usr/local/etc/rc.d/courier-imap-pop3d start
```

Для автоматического запуска добавьте в /etc/rc.conf следующую строку:

```
courier_imap_pop3d_enable="YES"
```

27.4. Установка и настройка SMTP-сервера

Установим Postfix из портов:

```
# cd /usr/ports/mail/postfix
# make install clean
```

При сборке Postfix нужно включить следующие опции: PCRE, TLS, MYSQL, VDA. При установке программы (уже после ее сборки) вам будут заданы несколько вопросов. Ответьте на них по своему усмотрению, но обязательно нужно ответить y на следующие вопросы:

```
You need user "postfix" added to group "mail".
Would you like me to add it [y]? y

Would you like to activate Postfix in /etc/mail/mailer.conf [n]? y
```

Ответив y на первый вопрос, вы добавите пользователя postfix в группу mail. От имени пользователя postfix будет запускаться наш MTA, поэтому не добавить его в эту группу нельзя. Ответ y на второй вопрос «пропишет» postfix в файле /etc/mail/mailer.conf. По умолчанию там прописан MTA sendmail. Если вы ответите n на второй вопрос, то вам вручную придется прописать пути к программе postfix в этом файле.

Настало время отредактировать файл конфигурации /usr/local/etc/postfix/main.cf. По сравнению с конфигурационным файлом sendmail файл postfix/main.cf не очень большой, но приводить его полностью не хочется. Вам нужно обратить внимание на следующие параметры:

```
# при установке программы был создан пользователь postfix, лучше запускать
# MTA от имени этого пользователя, а не от пользователя root
mail_owner = postfix

# Имя нашего почтовика
myhostname = mail.firma.com

# Наш домен:
mydomain = firma.com

# Прослушивать все интерфейсы
inet_interfaces = all

# Список сетей, узлы которых могут передавать почту через наш сервер
# Укажите только свою сеть, чтобы никто другой не смог отправлять
# почту через наш MTA, например:
mynetworks = 192.168.1.0/24 127.0.0.0/8

# Отключаем команду VRFY, позволяющую проверить существование
# определенного ящика, следовательно, никто не сможет получить
# список пользователей нашего сервера. Эдакая дополнительная
# защита от спама
disable_vrfy_command = yes

# Максимальное количество ошибок почтовой программы пользователя
# По превышении этого числа соединение будет разорвано
smtpd_hard_error_limit = 5
```

```
# Разрешаем поддержку устаревших SMTP-клиентов,
# например, outlook express 4 и MicroSoft Exchange version 5.0.
# Старые программы используют старую версию команды AUTH
broken_sasl_auth_clients = yes

# Анонимная аутентификация запрещена
smtpd_sasl_security_options = noanonymous

# Параметры квот
virtual_mailbox_limit_maps = mysql:$base/mysqlLookupMaps/quota.conf
virtual_maildir_extended=yes
virtual_mailbox_limit_override=yes
virtual_create_maildirsize = yes
virtual_overquota_bounce = yes
# Сообщение при превышении лимита почтового ящика
virtual_maildir_limit_message="Sorry, the user's mailbox is full"

# Максимальный размер письма - 10 Мбайт
message_size_limit = 10485760
```

По сути, postfix уже настроен. Далее нужно отредактировать ряд файлов, относящихся к нашей запланированной конфигурации, а не самому MTA. Начнем с файла /usr/local/lib/sasl2/smtpd.conf. По умолчанию он уже создан, но его нужно отредактировать, как показано в листинге 27.3.

Листинг 27.3. Файл /usr/local/lib/sasl2/smtpd.conf

```
pwcheck_method: auxprop
mech_list: PLAIN LOGIN CRAM-MD5
auxprop_plugin: sql
sql_engine: mysql
sql_usessl: yes
sql_hostnames: localhost
sql_user: postfix
sql_passwd: postfix
sql_database: postfix
sql_select: select password from mailbox where username = '%u@%r'
log_level: 3
```

Здесь мы указываем параметры аутентификации. Мы используем MySQL-сервер localhost, имя пользователя postfix, с паролем postfix, база данных тоже называется postfix. При отсутствии этих объектов (базы данных и пользователя) их нужно создать.

Создадим базу данных псевдонимов:

```
# /usr/local/bin/newaliases
```

После этого нужно создать еще несколько файлов и один служебный каталог:

```
# touch /usr/local/etc/postfix/hello_access /usr/local/etc/postfix/sender_access
# touch /usr/local/etc/postfix/recipient_access /usr/local/etc/postfix/client_access
# postmap /usr/local/etc/postfix/hello_access
# postmap /usr/local/etc/postfix/sender_access
# postmap /usr/local/etc/postfix/recipient_access
# postmap /usr/local/etc/postfix/client_access
# mkdir /usr/local/etc/postfix/mysqlLookupMaps
```

В каталоге mysqlLookupMaps нужно создать несколько файлов. Чтобы не оформлять каждый из этих файлов в отдельный листинг, рассмотрим один большой листинг, где жирным выделены имена файлов, после которых следуют строки, которые нужно добавить в тот или иной файл.

Листинг 27.4. Файлы из каталога mysqlLookupMaps

```
/usr/local/etc/postfix/mysqlLookupMaps/alias.conf
user = postfix
password = postfix
hosts = localhost
dbname = postfix
table = alias
select_field = goto
where_field = address

/usr/local/etc/postfix/mysqlLookupMaps/domain.conf
user = postfix
password = postfix
hosts = localhost
dbname = postfix
table = domain
select_field = domain
where_field = domain
additional_conditions = and active = '1' and backupmx = '0'

/usr/local/etc/postfix/mysqlLookupMaps/mailbox.conf
user = postfix
password = postfix
hosts = localhost
dbname = postfix
table = mailbox
select_field = maildir
where_field = username
additional_conditions = and active = '1'

/usr/local/etc/postfix/mysqlLookupMaps/quota.conf
user = postfix
password = postfix
hosts = localhost
dbname = postfix
table = mailbox
select_field = quota
where_field = username
additional_conditions = and active = '1'

/usr/local/etc/postfix/mysqlLookupMaps/sender.conf
user = postfix
password = postfix
hosts = localhost
dbname = postfix
table = mailbox
select_field = username
where_field = username
additional_conditions = and active = '1
```

Понимаю, что процесс создания почтовика затянулся, поэтому на данный момент у меня есть две новости. Хорошая: почти все сделано, остались мелочи. Плохая: придется еще потрудиться.

Нам нужно создать каталог для почты, установить к нему права доступа, а также создать пользователя virtual и группу с таким же именем. Сначала изменим права доступа к каталогу mysqlLookupMaps:

```
# chown -R root:postfix /usr/local/etc/postfix/mysqlLookupMaps/
# chmod 440 /usr/local/etc/postfix/mysqlLookupMaps/*.conf
# chmod 550 /usr/local/etc/postfix/mysqlLookupMaps/
```

Добавим пользователя и группу virtual:

```
# pw group add virtual -g 3333
# pw user add virtual -g virtual -s /sbin/nologin -u 3333
```

Помните, в файле authmysqlrc мы указывали UID пользователя и GID группы? Так вот, сейчас мы создали пользователя и группу с указанными идентификаторами. Установим права доступа к каталогу с почтой:

```
# mkdir /var/spool/mail
# chown virtual:virtual /var/spool/mail/
# chmod 740 /var/spool/mail/
```

Обеспечим автоматический запуск postfix. Добавьте строку postfix_enable="YES" в /etc/rc.conf. Если до этого у вас была установлена программа sendmail, перед только что добавленной строкой добавьте вот эту строку:

```
sendmail_enable="NO"
```

Запустите postfix:

```
# /usr/local/etc/rc.d/postfix start
postfix/postfix-script: starting the Postfix mail system
```

Вот на этом действительно все. Осталось только установить веб-интерфейс для управления postfix:

```
# cd /usr/ports/mail/postfixadmin
# make install clean
```

Для работы интерфейса PostfixAdmin нужен полноценный веб-сервер с поддержкой PHP. PHP, в свою очередь, должен поддерживать GD и MySQL. Такой сервер мы настроили в предыдущей главе, поэтому все, что нам нужно сделать, — это установить PostfixAdmin и отредактировать файл конфигурации Apache. Откройте файл /usr/local/etc/apache22/httpd.conf и добавьте в него строки:

```
<Directory "/usr/local/www/postfixadmin/">
    Options none
    AllowOverride Limit
    Order Deny,Allow
    Deny from all
    Allow from 192.168.1.100
</Directory>
```

IP-адрес 192.168.1.100 — это IP-адрес рабочей станции администратора (то есть вашей рабочей станции). Запустить PostfixAdmin можно будет только с этой ра-

бочей станции. После сохранения файла конфигурации Apache перезапустите веб-сервер.

Создайте служебные таблицы, необходимые для нашей конфигурации:

```
# cd /usr/local/www/postfixadmin
# mysql -u root -p < DATABASE_MYSQL.TXT
```

Отредактируйте файл конфигурации PostfixAdmin (`/usr/local/www/postfixadmin/config.inc.php`), найдите следующие параметры и установите их так:

```
$CONF['database_type'] = 'mysql';
$CONF['database_host'] = 'localhost';
$CONF['database_user'] = 'postfixadmin';
$CONF['database_password'] = 'postfixadmin';
$CONF['database_name'] = 'postfix';
$CONF['database_prefix'] = "";
…
…
$CONF['configured'] = true;
```

Настройки, я думаю, понятны. Мы задаем имя MySQL-сервера, имя пользователя, пароль и базу данных. Осталось запустить сценарий `setup.php` (это нужно сделать с компьютера 192.168.1.100). Откройте этот сценарий в любом браузере:

http://*адрес-сервера*/postfixadmin/setup.php

После настройки удалите сценарий `setup.php`, для запуска PostfixAdmin введите URL:

http://*адрес-сервера*/postfixadmin/

Вот на этом действительно все. Теперь в вашем распоряжении есть полноценный почтовый сервер, который, к тому же, легок в администрировании благодаря веб-интерфейсу PostfixAdmin.

Шлюз своими руками

28

28.1. Что такое шлюз?

Шлюз — это компьютер, предоставляющий другим компьютерам сети доступ к Интернету. Как правило, у шлюза есть как минимум два сетевых интерфейса. Один сетевой интерфейс служит для подключения к Интернету, а другой — для подключения к локальной сети. Сам же шлюз занимается «перебрасыванием» пакетов с локального на внешний интерфейс и обратно[1].

Помимо этого шлюз выполняет еще одну очень важную функцию — NAT (Network Address Translation) — преобразование сетевых адресов. Дело в том, что в большинстве случаев в нашей локальной сети будут использоваться локальные адреса (например, 192.168.1.0, 10.0.0.0), но с такими IP-адресами не работают маршрутизаторы Интернета. Даже если мы отправим в сеть пакет с локальным адресом, он будет отброшен первым же маршрутизатором. Данные IP-адреса должны использоваться только в локальной сети.

Как же тогда компьютеру с локальным адресом, скажем, 192.168.1.23, связаться с компьютером, у которого реальный IP-адрес? А вот для этого используется NAT. У нашего шлюза есть внешний интерфейс, через который осуществляется соединение с Интернетом. Этому интерфейсу присвоен реальный IP-адрес, скажем,

[1] С терминологией в этой области есть определенные проблемы. С одной стороны, «шлюз» (gateway) — аппаратно-программный комплекс, адрес которого указывается всем остальным узлам сети как адрес, на который следует пересылать обращения к узлам за ее пределами («чужим»). С другой — аналогичную функцию (пересылка данных между своими интерфейсами, в зависимости от их адресатов) выполняет «маршрутизатор» (router) — тоже являющийся аппаратно-программным комплексом (АПК). С третьей — «межсетевой экран» (firewall, «файервол», «брандмауэр») — АПК, основным назначением которого является предотвращение нежелательного взаимодействия между различными сетями; для выполнения своих функций также производит анализ проходящих через него данных, в том числе и по адресам отправителей и получателей. При этом все три АПК для выполнения своих функций могут как использовать трансляцию адресов, так и не использовать. И все три АПК могут как представлять собой полностью автономные устройства (отдельные узлы сети), так и быть реализованными программно на одном узле. Причем программная реализация может быть как отдельной для каждой задачи, так и одним пакетом программ. Поэтому, часто все три термина рассматриваются как синонимы, что далеко не всегда верно.

94.94.94.94. Иначе нельзя — ведь без реального адреса нельзя было бы установить соединение с серверами Интернета.

Задача NAT заключается в подмене IP-адреса. Когда компьютер с адресом 192.168.1.23 отправляет пакет какому-нибудь серверу, например www.mail.ru, то шлюз делает подмену IP-адреса — как будто он был отправлен от имени узла с адресом 94.94.94.94. Удаленный компьютер, то есть www.mail.ru, «думает», что пакет пришел от узла с адресом 94.94.94.94. Ответ тоже будет отправлен на 94.94.94.94. Получив ответ от www.mail.ru, наш шлюз перезапишет получателя этого узла и отправит его узлу 192.168.1.23. В результате узел 192.168.1.23 даже не заметит, что пакет был отправлен не ему, а сначала узлу 94.94.94.94 — он будет «думать», что общается с www.mail.ru напрямую. В этом и заключается задача NAT.

Итак, мы уже знаем две функции шлюза — перенаправление пакетов (IP Forwarding) и NAT. Есть и еще одна, не менее важная фильтрация пакетов, позволяющая защитить нашу сеть от вмешательства извне. Как правило, шлюз запрещает другим узлам Интернета доступ к нашей локальной сети. Реализуется это с помощью фильтрации пакетов.

В Linux шлюз настраивается с помощью iptables, во FreeBSD — с помощью ipfw. В этой главе мы будем строить свой шлюз на базе FreeBSD — ведь мы договорились, что при настройке сервера будем больше внимания уделять FreeBSD. Если у вас на сервере установлен Linux, то с iptables вы можете ознакомиться в следующем подробнейшем руководстве (на русском языке): http://www.opennet.ru/docs/RUS/iptables/. Выбор FreeBSD в качестве героя этой главы обусловлен еще одним фактором: в BSD шлюз настраивается сложнее, чем в Linux. В Linux вам не нужно перекомпилировать ядро, разбираться с конфигурационными файлами. Нужно только прочитать руководство по iptables. А если у вас небольшая локальная сеть, то с поставленной задачей справится графическая программа-оболочка firestarter, которая есть во многих дистрибутивах. С помощью этой программы вы превратите свой компьютер в шлюз за пару минут.

28.2. Перекомпиляция ядра FreeBSD

В этой книге не рассматривалась перекомпиляция ядра. До этого момента. Вы не сможете настроить полноценный шлюз, если не перекомпилируете ядро. Сначала разберемся, какие опции нужно добавить в файл конфигурации ядра, а затем уже поговорим о самой перекомпиляции. В файл конфигурации ядра нужно добавить следующие опции:

```
options IPFIREWALL
options IPFIREWALL_DEFAULT_TO_ACCEPT
options IPFIREWALL_FORWARD
options IPDIVERT
```

Первая опция включает сам межсетевой экран (он же брандмауэр, он же firewall — программное обеспечение, необходимое для организации шлюза). Вторая опция устанавливает политику по умолчанию: разрешен прием всех пакетов. Третья опция разрешает использовать перенаправление пакетов, что пригодится

также при построении прозрачного прокси. Последняя опция включает поддержку NAT.

А вот теперь пора разобраться с компиляцией ядра. Загляните в каталог /usr/src/sys. Если там пусто, значит, исходные коды ядра не установлены. Установить «исходники» ядра просто — вставьте дистрибутивный DVD (с которого вы установили FreeBSD) и введите три команды:

```
# mount /cdrom
# cd /cdrom/8.1-RELEASE/src
# ./install.sh
```

ПРИМЕЧАНИЕ

Если у вас другая версия FreeBSD, то вторую команду нужно немного исправить.

Подождите, пока будут извлечены все файлы. После этого перейдите в каталог /usr/src/sys/i386/conf. В этом каталоге хранятся файлы конфигурации ядра для архитектуры процессора i386. Если у вас другая архитектура, тогда нужно перейти в каталог /usr/src/sys/<название архитектуры>/conf.

В каталоге conf находится файл GENERIC — это базовая конфигурация ядра. Скопируйте этот файл:

```
# cp GENERIC gateway
```

Мы скопировали файл GENERIC в файл gateway, с которым и будем работать. В этот файл и нужно добавить те самые строки.

После этого нужно перекомпилировать само ядро:

```
# cd /usr/src
# make buildkernel TARGET_ARCH=i386 KERNCONF=gateway
# make installkernel TARGET_ARCH=i386 KERNCONF=gateway
```

Первый параметр (TARGET_ARCH) задает название архитектуры, нужно указать ваше название. Второй параметр — название файла конфигурации, в нашем случае — gateway. После того как завершится выполнение последней команды, перезагрузите компьютер.

ПРИМЕЧАНИЕ

Подробно процесс компиляции ядра рассмотрен по адресу:

http://www.freebsd.org/doc/ru/books/handbook/kernelconfig-building.html

28.3. Редактирование файла /etc/rc.conf

После перезагрузки нужно отредактировать файл /etc/rc.conf, чтобы включить все, что относится к шлюзу, и настроить сетевые интерфейсы. Пусть у нас есть два сетевых интерфейса: rl0 с адресом 192.168.1.1 и rl1 с адресом 94.94.94.94. Понятно, что первый интерфейс является внутренним (подключен к локальной сети), а второй — внешним (подключен к Интернету).

Для их настройки нужно добавить в rc.conf примерно такие строки (адреса и сетевые маски у вас будут другими):

```
ifconfig_rl0="inet 192.168.1.1 netmask 255.255.255.0"
ifconfig_rl1="inet 94.94.94.94 netmask 255.255.255.0"
```

Затем добавляем следующие строки:

```
defaultrouter="94.94.94.1"
hostname="gate.company.com"
gateway_enable="YES"
firewall_enable="YES"
firewall_type="OPEN"
natd_enable="YES"
natd_interface="rl1"
```

В первой строке указываем IP-адрес шлюза по умолчанию — это шлюз провайдера. Вторая строка задает имя нашего узла — шлюза. Третья строка превращает наш компьютер в шлюз. Четвертая запускает межсетевой экран (брандмауэр). Брандмауэр нужен для фильтрации пакетов. Теоретически можно обойтись и без него, но не стоит этого делать.

Четвертая строка задает тип брандмауэра. Об этом мы поговорим чуть позже, поскольку это заслуживает отдельного разговора. Следующие две строки разрешают NAT и заставляют демон natd прослушивать внешний интерфейс.

А теперь поговорим о типах брандмауэра. Тип OPEN означает, что брандмауэр будет пропускать все пакеты, то есть фильтрации как таковой не будет. CLOSED означает закрытый брандмауэр — будут отклоняться все пакеты, кроме пакетов, пришедших с локального интерфейса lo0. По умолчанию используется тип UNKNOWN. Что будет происходить с пакетами, зависит от политики, выбранной при конфигурации ядра. Мы выбрали политику IPFIREWALL_DEFAULT_TO_ACCEPT, поэтому наш брандмауэр будет пропускать все пакеты. Есть еще политика CLIENT, но она подходит не для шлюза, а просто для защиты клиентского компьютера — она разрешает исходящие соединения, но блокирует входящие соединения к компьютеру. Все эти политики описываются в файле /etc/rc.firewall. Вместо /etc/rc.firewall можно указать другой сценарий:

```
firewall_script="/etc/rules"
```

Файл /etc/rules устанавливает набор правил нашего брандмауэра. В листинге 28.1 приводится пример этого файла.

Листинг 28.1. Файл /etc/rules

```
#!/bin/sh
/sbin/ipfw -f flush
/sbin/ipfw add pass all from any to any via lo0
/sbin/ipfw add deny all from any to 127.0.0.0/8
/sbin/ipfw add deny icmp from any to any in icmptype 5,9,13,14,15,16,17
/sbin/ipfw add deny icmp from any to any frag
/sbin/ipfw add deny tcp from any to any not established tcpflags fin
/sbin/ipfw add deny tcp from any to any tcpflags fin,syn,rst,psh,ack,urg
/sbin/ipfw add deny tcp from any to any tcpflags !fin,!syn,!rst,!psh,!ack,!urg
/sbin/ipfw add deny udp from any 137-139 to any via rl0
/sbin/ipfw add deny udp from any to any 137-139 via rl0
```

продолжение ⇩

Листинг 28.1 *(продолжение)*

```
/sbin/ipfw add divert natd ip from 192.168.1.0:255.255.255.0 to any out xmit rl1
/sbin/ipfw add divert natd ip from any to 94.94.94.94
/sbin/ipfw add deny all from 192.168.1.0/24 to not 192.168.1.0/24 80,20,21,443
/sbin/ipfw add allow all from any to any
```

Это простейший набор правил, блокирущий фрагментированные пакеты, запрещающий подключения извне к компьютерам нашей сети и т. д. Пример вполне рабочий, вам нужно подставить только свои IP-адреса. Узнать о формате команды `ipfw` можно в `man ipfw`.

По сути, наша задача выполнена. Вы можете настроить прокси-сервер Squid, а поскольку мы разрешили перенаправление пакетов, то теперь вы можете настроить прозрачный прокси-сервер.

V

Будни системного администратора

Подсчет трафика

29

29.1. Постановка задачи

Рано или поздно перед администратором встает задача подсчета трафика. Раньше этим заставляли заниматься каждого администратора, поскольку услуги доступа к Интернету были довольно дорогими. Сейчас цены стали дешевле, а на потребляемый трафик уже никто не обращает внимания — все провайдеры давно перешли на так называемые безлимитные пакеты, которые отличаются только скоростью передачи данных.

Так что в наши дни администратор «заморачивается» с созданием системы подсчета трафика, только если у него достаточно свободного времени. Нужно же чем-то занять и себя, и сервер.

Существует множество способов подсчета трафика во FreeBSD. Можно организовать собственную систему подсчета трафика — это уж если слишком много времени и у вас есть тяга к творчеству, а также навыки программирования. А можно установить уже готовую систему подсчета трафика, настроить ее и использовать.

В этой главе мы рассмотрим создание системы подсчета трафика на базе trafd и использование уже готовой системы darkstat.

29.2. Использование trafd

Будем считать, что шлюз на базе FreeBSD уже успешно настроен. После настройки шлюза нам понадобится утилита `trafshow`, которую можно установить из портов:

```
# cd /usr/ports/net/trafshow
# make install
```

Мы можем просмотреть весь трафик, проходящий через интерфейс, командой:

```
# trafshow -i <имя интерфейса>
```

Например:

```
# trafshow -i em0
```

В справочном руководстве по программе trafshow (man trafshow) вы найдете список опций, с помощью которых можно задать фильтры. Например, следующая команда выводит весь трафик, проходящий через em0 в сеть 192.168.1.0, при этом опция -n указывает, что в выводе таблицы трафика будут использоваться только IP-адреса, а не доменные имена:

```
# trafshow -i em0 -n dst net 192.168.1.0 mask 255.255.255.0
```

Как мы знаем, у шлюза есть как минимум два сетевых интерфейса — внутренний и внешний. Для получения достоверной картины расхода трафика нужно просматривать именно внутренний интерфейс, а не внешний. Ведь очень часто используется система NAT, поэтому на внешнем интерфейсе все внутренние адреса «собраны» в один — во внешний адрес. Именно поэтому нужно анализировать внутренний интерфейс, а не внешний.

Теперь установим демон trafd, который будет заниматься сбором статистики:

```
# cd /usr/ports/net/trafd
# make install
```

Демон trafd собирает трафик на указанном интерфейсе. Собранная статистика помещается в каталог /var/trafd. Запустим сбор статистики на интерфейсе em0:

```
# /usr/local/bin/trafd -i em0
```

Для автоматического запуска trafd пропишите эту строчку в /etc/rc.local. Но при перезагрузке информация о трафике будет теряться, поэтому нужно использовать утилиту trafsave. Для периодического сохранения статистики нужно прописать вызов этой команды в расписание /etc/crontab:

```
0 0,4,8,12,16,20 * * 1-7 /usr/local/bin/trafsave
```

Статистика интерфейса em0 будет сохранена в файле /var/trafd/em0. Периодически (раз в месяц) нужно обнулять этот файл, а его копию сохранять под другим именем. Это неудобно делать вручную, поэтому нужно использовать программу rotate, которую тоже нужно будет прописать в crontab:

```
rotate /var/trafd/<интерфейс> `date -v-1m +%b`
```

Если программа rotate у вас не установлена, ее нужно установить:

```
cd /usr/ports/sysutils/rotate
make install
```

Осталось только собрать отчет. Для построения отчетов используется программа traflog. Вот примеры вызова этой программы (назначение параметров у нее такое же, как у программы trafshow):

```
# traflog -n -i em0
# traflog -i em0 -n from all to 192.168.1.0 mask 255.255.255.0
```

29.3. Использование darkstat

Теперь рассмотрим другую систему статистики darkstat. Эта система позволяет строить красивые графические графики (это не «масло масляное» — некоторые системы строят графики, используя элементы псевдографики, которые менее

привлекательны). Ее проще установить не из портов, а из пакетов, поэтому введите команду:

```
# pkg_add -r darkstat
```

Затем добавьте в /etc/rc.conf следующие строки:

```
darkstat_enable="YES"
darkstat_interface="em0"
```

Первая строка — запускает демон darkstat, а вторая указывает интерфейс, который нужно прослушивать. Давайте сразу просмотрим графики darkstat, но чтобы на графиках что-то было, нужно сгенерировать какой-то трафик, загрузив большой файл:

```
# fetch <имя файла>
```

Можете использовать образ FreeBSD. Демон darkstat работает на 667-м порту, потому запустите браузер для просмотра статистики и введите URL:

```
http://имя_узла:667
```

На моем шлюзе установлен графический интерфейс, поэтому я могу запустить Firefox прямо на шлюзе и обратиться к darkstat так: http://localhost:667. На рисунках 29.1 и 29.2 система darkstat изображена в действии

Мы проверили, что система работает. Теперь нужно обеспечить загрузку и сохранение статистики. Для этого в /etc/rc.conf добавьте строку, описывающую параметры darkstat:

```
darkstat_flags="--import iem0.db --export iem0.db"
```

Раз в месяц нужно обеспечить ротацию файла iem0.db с помощью утилиты rotate, вызов которой нужно добавить в расписание crontab.

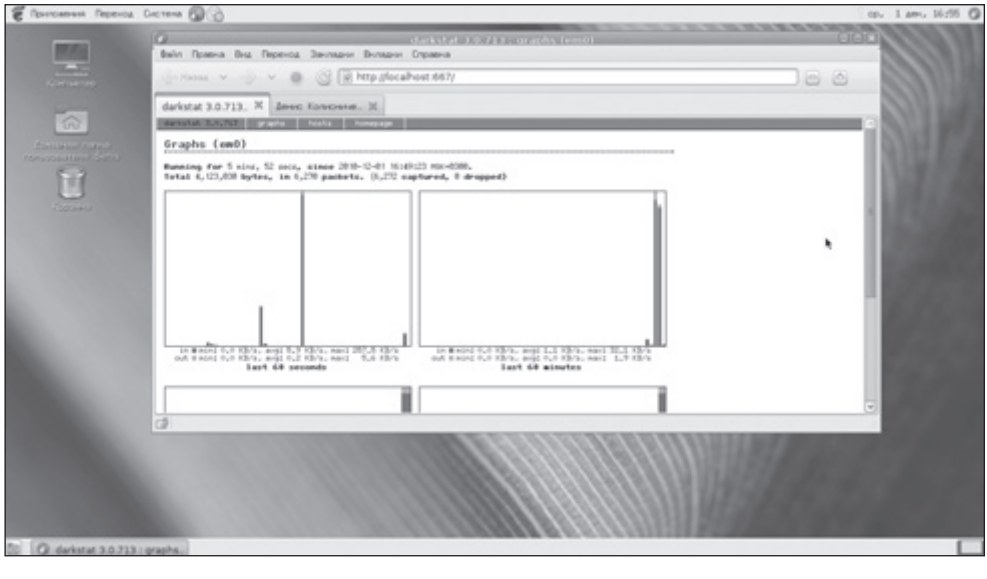

Рис. 29.1. Графики загрузки интерфейса

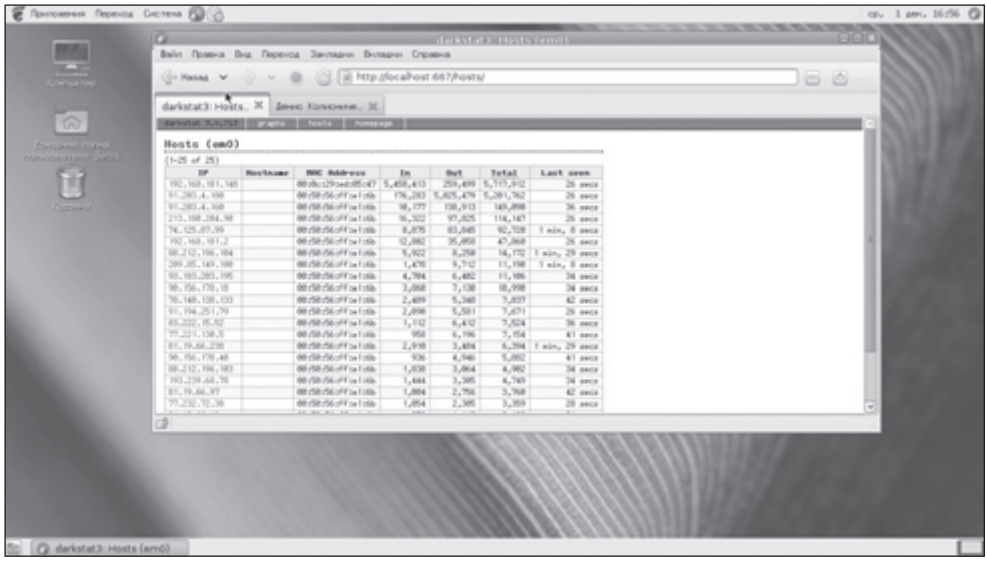

Рис. 29.2. Статистика по узлам

Сетевой сканер nmap

30

30.1. Назначение nmap

Программа nmap — настоящая кинозвезда. Наверное, нет ни одного читателя этой книги, который бы не смотрел фильм «Матрица». В первой части этого фильма можно увидеть в работе сетевой сканер nmap:

http://www.youtube.com/watch?v=0TJuipCrjZQ

Наконец-то в фильме можно увидеть узнаваемую программу. А то смотришь американские фильмы и видишь в них непонятные операционные системы. Такое впечатление, что они создавались специально для фильма, иначе нельзя объяснить, откуда они взялись. А тут реальная программа, с которой может поработать любой желающий.

Но в этой книге nmap представлена не потому, что она «засветилась» в «Матрице», а вполне заслуженно. Программа nmap — это мощный сетевой сканер, позволяющий сканировать как отдельные узлы, так и целые сети. Программа поддерживает много разных методов сканирования — TCP connect, TCP SYN, ICMP и т. д. Справочная страница nmap на русском языке доступна на моем сайте:

http://dkws.org.ua/man/nmap.html

Мне не хочется полностью переписывать справочное руководство по nmap. Это пустая трата и своего времени, и ваших денег — поскольку справочную страницу вы можете прочитать самостоятельно (тем более, она на русском языке). Поэтому в данной главе мы рассмотрим только несколько практических примеров использования nmap.

Как уже было отмечено, сканер можно использовать для сканирования целой сети. Кроме этого у сканера есть даже «диагностические функции», которые можно использовать для аудита сети: определение наличия брандмауэров, вычисление времени задержки, определение операционной системы и т. д.

В результате работы сканера вы увидите список просканированных портов удаленного компьютера. Сканер выводит номер и состояние порта, используемый портом протокол и, по возможности, определяет службу, использующую

порт. Порт может быть открыт, закрыт брандмауэром или попросту не использоваться.

Программа nmap часто входит в состав дистрибутивов Linux. В этом случае ее можно установить прямо из репозитория, например:

```
sudo apt-get install nmap
```

На сайте разработчиков можно также скачать nmap, в том числе исходные коды и Windows-версию:

http://nmap.org/download.html

Не беспокойтесь, есть также версия и для FreeBSD.

30.2. Использование nmap

Запускать nmap нужно от имени root, поэтому перед запуском или войдите в систему как root или введите команду su (можно также использовать sudo, если в вашей системе она установлена и настроена), чтобы получить права root. Формат вызова программы следующий:

```
# nmap опции цель
```

Довольно интересная опция — это определение операционной системы. Давайте посмотрим, как сканер с ней справляется. Для определения операционной системы используется опция -O:

```
# nmap -O <IP-адрес или имя узла>
```

Я просканировал сервер своей сети, работающий под управлением FreeBSD. Вот какой отчет мне предоставил nmap:

```
Starting Nmap 5.21 ( http://nmap.org ) at 2010-10-03 13:39 EST
Nmap scan report for 10.129.0.1

Host is up (0.0013s latency).
Not shown: 808 closed ports, 184 filtered ports
PORT    STATE SERVICE
22/tcp  open  ssh
25/tcp  open  smtp
53/tcp  open  domain
80/tcp  open  http
110/tcp open  pop3
143/tcp open  imap
179/tcp open  bgp
995/tcp open  pop3s
MAC Address: xx:xx:xx:xx:xx:xx (Intel Corporate)
Device type: general purpose
Running: FreeBSD 7.X|8.X
OS details: FreeBSD 7.0-RELEASE-p1 – 8.0-CURRENT, FreeBSD 7.2-RELEASE
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 9.22 seconds
```

Как видите, nmap отлично справилась с поставленной задачей — она определила операционную систему и не ошиблась. Даже предположила номер версии (сервер работал под управлением FreeBSD 7.2).

Как видно из вывода nmap, на сервере открыты порты 22, 25, 53, 80, 110, 143, 179 и 995. Поскольку я находился в одном сегменте сети с сервером, nmap даже сообщила его MAC-адрес.

Помните, что сканировать желательно только свои узлы, чтобы потом у вас не возникло проблем с законом. Если же хотите просто потренироваться, вы всегда можете просканировать тестовый сервер — scanme.nmap.org.

Чтобы просканировать сразу несколько узлов, можно указать диапазон IP-адресов или имена/IP-адреса компьютеров через пробел, например:

```
# nmap 10.1.1.1-250
# nmap server1 server2
# nmap server1 10.1.1.7
```

Если не указывать опции, то будет выполнено обычное сканирование портов. Вывод будет примерно такой:

```
Starting Nmap 5.21 ( http://nmap.org ) at 2010-10-03 13:59 EST
Nmap scan report for 10.129.0.1Host is up (0.0039s latency).Not shown: 808 closed ports,
184 filtered portsPORT    STATE SERVICE22/tcp  open  ssh25/tcp  open  smtp53/tcp  open
domain80/tcp  open  http110/tcp open  pop3143/tcp open  imap179/tcp open  bgp995/tcp open
pop3sMAC Address: XX:XX:XX:XX:XX:XX (Intel Corporate)
Nmap done: 1 IP address (1 host up) scanned in 6.23 seconds
```

Можно узнать подробное имя службы, для этого используется опция -sV. Сканер подробно расскажет о каждой сетевой службе, запущенной на удаленном компьютере:

```
Nmap scan report for 10.129.0.1

Host is up (0.0015s latency).
Not shown: 838 closed ports, 154 filtered ports
PORT    STATE SERVICE    VERSION
22/tcp  open  ssh        OpenSSH 5.1p1 (FreeBSD 20080901; protocol 2.0)
25/tcp  open  smtp       Exim smtpd 4.71
53/tcp  open  domain     ISC BIND 9.4.3-P2
80/tcp  open  http       Apache httpd 2.2.13 ((FreeBSD) mod_ssl/2.2.13 OpenSSL/0.9.8e DAV/2
PHP/5.2.11 with Suhosin-Patch)
110/tcp open  pop3       Dovecot pop3d
143/tcp open  imap       Dovecot imapd
179/tcp open  tcpwrapped
995/tcp open  ssl/pop3   Dovecot pop3d
```

Еще есть полезная опция -PN, позволяющая узнать, активен ли узел. Иногда, если узел надежно скрыт брандмауэром, nmap сообщает, что узел недоступен. Тогда можно использовать опцию -PN, чтобы узнать, запущен ли узел.

Опция -sP позволяет определить, какие узлы сети являются доступными (включены). Полезно для мониторинга всей сети, например:

```
# nmap -sP диапазон-адресов
```

По каждому узлу выдается краткий отчет — включен ли он и MAC-адрес узла (свои MAC-адреса я скрыл):

```
Starting Nmap 5.21 ( http://nmap.org ) at 2010-10-03 14:15 EST
```

```
Nmap scan report for 10.129.0.1
Host is up (0.0010s latency).
MAC Address: xx:xx:xx:xx:xx:xx (Intel Corporate)
Nmap scan report for 10.129.0.4
Host is up (0.018s latency).
MAC Address: xx:xx:xx:xx:xx:xx (Digital China (Shanghai) Networks)
Nmap scan report for 10.129.0.5
Host is up (0.0030s latency).
MAC Address: xx:xx:xx:xx:xx:xx (Digital China (Shanghai) Networks)
...
```

30.3. Windows-версия nmap

Windows-версия сканера nmap поставляется вместе с графической оболочкой Zenmap (рис. 30.1). От этого использование nmap легче не станет — все равно нужно знать опции nmap (ну или воспользоваться профилем, но набор профилей ограничен). Зато оболочка хранит список просканированных узлов, и в любой момент вы можете просмотреть информацию о ранее просканированном узле без повторного сканирования.

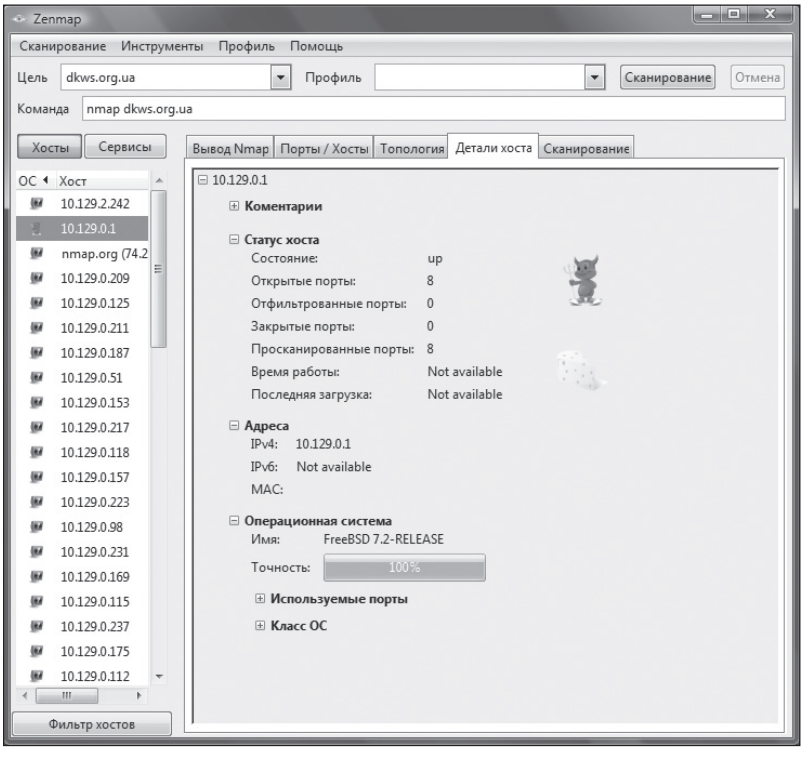

Рис. 30.1. Сканер nmap для Windows

Антивирусная проверка веб-трафика 31

31.1. Различные способы проверки трафика

Весь веб-трафик проходит через шлюз, на котором обычно запущен прокси — для ускорения доступа к Интернету (кэширования страниц) и экономии трафика. Трафик может быть как полезным, так и не очень, даже опасным. Вместе с загруженной веб-страницей на компьютер пользователей могут попасть вирусы, распространяющиеся через WWW. Как показывает практика, такие вирусы не менее опасны, чем вирусы, распространяющиеся через сменные носители информации, а в последнее время сетевые вирусы наиболее популярны среди «вирусописателей». Так что сегодня «подхватить» сетевой вирус более реально, чем вирус, передающийся с флешки на флешку[1].

А поскольку весь трафик будет проходить через прокси, ничто не мешает нам проверить его на наличие вирусов. Конечно, такая проверка не освобождает вас от установки антивируса на рабочие станции — ведь есть еще много способов инфицирования компьютера — через те же флешки, по почте и т. д.

Существует несколько способов организации взаимодействия прокси-сервера `Squid` и антивируса. Первый заключается в задействовании протокола ICAP, используемого прокси-серверами для фильтрации трафика. Но этот способ очень сложно настроить (да и нет никакой гарантии, что в результате всех произведенных действий он заработает), поэтому мы его применять не будем. Второй способ заключается в использовании специального прокси `HAVP` (http AntiVirus Proxy), но этот способ, наоборот, слишком легкий в настройке и сводится к установке Squid, ClamAV и HAVP. Потом нужно только отредактировать конфигурационный файл Squid — и все заработает. Описание этого способа вы без проблем найдете в Интернете. Третий способ заключается в использовании редиректоров. Он более сложный в настройке, но демонстрирует использование редиректоров Squid, поэтому мы будем рассматривать именно его. Возможно, полученные знания о редиректорах позволят вам создать совершенно другую конфигурацию (не для

[1] К сожалению, уже встречаются и гибриды, способные распространяться несколькими способами одновременно, и не только этими. И на флешках для некоторых операционных систем может представлять опасность не только autorun.inf, но и казалось бы «безобидные» файлы ярлыков *.lnk.

проверки трафика), поскольку возможности редиректоров антивирусной проверкой не ограничиваются.

31.2. Необходимое программное обеспечение

Для реализации поставленной задачи нам нужен прокси-сервер Squid, расширение SquidGuard, веб-сервер Apache, скрипт viralator и собственно сам антивирус ClamAV. ClamAV — лучший бесплатный антивирус для UNIX, именно поэтому мы выбрали именно его.

Все программное обеспечение мы будем устанавливать из портов, поскольку только в этом случае можно включить все необходимые нам опции, чего нельзя сделать при установке программ из пакетов.

Учитывая, что настройка Squid и Apache уже рассматривалась в этой книге (в главах 24 и 26 соответственно), мы не будем подробно описывать процесс установки этих сетевых сервисов. Будет рассмотрена лишь настройка, относящаяся к нашей конфигурации.

31.3. Установка Squid и SquidGuard

Если прокси-сервер еще не установлен, установим его:

```
# cd /usr/ports/www/squid
# make install clean
```

После этого отредактируем файл конфигурации /usr/local/etc/squid/squid.conf. Найдите и установите следующие директивы, как показано ниже:

```
http_port 3128
cache_dir ufs /usr/local/squid/cache 521 32 256
acl office src 192.168.1.0/24
http_access allow office
http_access deny all
cache_effective_user squid
cache_effective_group squid
visible_hostname {$hostname | $ip_address}
```

Вам нужно изменить только параметры кэша (см. главу 24) и адрес вашей сети. Затем нужно создать каталог для кэша /usr/local/squid/cache:

```
# cd  /usr/local/squid/
# mkdir cache
# chown squid:squid cache
# squid -z
```

Первая команда переходит в родительский каталог, вторая — создает каталог кэша, третья — устанавливает должны образом права доступа к каталогу, а последняя команда создает сам кэш.

Теперь запустим сам Squid:

```
# squid
```

О том, как организовать автоматическую загрузку Squid, было рассказано в главе 24, так что не забудьте это сделать.

Теперь установим SquidGuard:

```
# cd /usr/ports/www/squidguard
# make install clean
# touch /usr/local/etc/squid/squidGuard.conf
```

Последняя команда создает пустой файл конфигурации squidGuard.conf. В него нужно добавить следующие строки:

```
dbhome /usr/local/etc/squid/db
  logdir /var/log
  dest files {
  expressionlist proibidos/files
  }
  acl {

  default {
  pass !files all
  redirect
  http://192.168.1.1/cgi-bin/viralator.cgi?url=%u
   }
  }
```

Каталоги /usr/local/etc/squid/db и /usr/local/etc/squid/db/proibidos, как правило, отсутствуют. Их нужно создать вручную:

```
# mkdir /usr/local/etc/squid/db
# mkdir /usr/local/etc/squid/db/proibidos
```

После этого создайте в /usr/local/etc/squid/db/proibidos файл с названием files:

```
# ee  /usr/local/etc/squid/db/proibidos/files
```

В этот файл добавьте следующую строчку:

```
(\.exe$|\.bat$|\.bin$|\.reg$|\.sys$|\.avi$|\.mpg$|\.mpeg$)
```

В файл files добавляется регулярное выражение. Прокси-сервер разрешит загрузку любых файлов, которые не соответствуют этому выражению. В нашем случае запрещается загрузка файлов с расширениями .exe, .bat, .bin, .reg, .sys, .avi, .mpg, .mpeg[1]. Первые пять типов файлов могут быть потенциально опасными (содержать вирус), а последние три мы запретили для экономии трафика (размер таких файлов, как правило, большой).

[1] Перечень запрещенных расширений файлов несколько спорный. Во-первых, из-за него станет невозможным скачивать, например, бесплатные антивирусы для все еще остающихся в сети рабочих станций под управлением Windows (часто распространяемых как .exe-файлы), а во-вторых — никто не мешает злоумышленнику переименовать .exe-файл с вирусом в файл «любимая музыка системного адмнистратора.mp3». И еще хуже, если какой-нибудь .avi-файл (с демонстрацией работы закупаемого оборудования или чего-нибудь в этом роде) не сможет срочно скачать кто-нибудь из руководства предприятия — проблемы для системного администратора гарантированы. Лучше все проверять антивирусом, не делая предположений о формате и содержании файлов исходя лишь из их имен.

192.168.1.1 — это IP-адрес нашего веб-сервера, который нам предстоит настроить.

Теперь осталось подключить SquidGuard к Squid. Для этого добавим в squid.conf строки:

```
redirect_program /usr/local/bin/squidGuard
redirect_children 15
redirector_access deny localhost
redirector_access deny SSL_ports
```

Сохраните конфигурационный файл и перезагрузите Squid. Все, Squid полностью готов к работе. Далее нужно установить и настроить веб-сервер, установить viralator и ClamAV.

31.4. Установка и настройка Apache

Устанавливаем Apache 2.2 из портов:

```
# cd /usr/ports/www/apache22
# make install clean
```

Отредактируем его конфигурационный файл:

```
# ee /usr/local/etc/apache22/httpd.conf
```

Нужно найти и установить следующие директивы так:

```
Listen 192.168.1.1:80
ServerName 192.168.1.1
DocumentRoot  /usr/local/www
```

Подробно настройка Apache рассмотрена в главе 24. Чтобы изменения вступили в силу, перезагружаем Apache.

31.5. Установка ClamAV

Устанавливаем ClamAV из портов:

```
# cd /usr/ports/security/clamav
# make install clean
```

Обновляем антивирусные базы:

```
# /usr/local/etc/rc.d/clamav-clamd start
# freshclam
```

В /etc/rc.conf нужно добавить две строчки (первая — запуск демона clamd, вторая — автоматическое обновление ClamAV):

```
clamav_clamd_enable="YES"
clamav_freshclam_enable="YES"
```

31.6. Установка и настройка viralator

Скрипт viralator нужен для перевода запросов со squidGuard на антивирус ClamAV. Скачать viralator можно с сайта разработчиков: http://viralator.sourceforge.net/.

Распакуйте архив в каталог /usr:

```
# tar xfvz viralator-X.X.X.tar.gz
```

Появится каталог viralator-X.X.X, перейдите в нее и скопируйте файл viralator.cgi в каталог cgi-bin сервера, после нужно установить владельца для файла viralator.cgi:

```
# cp viralator.cgi /usr/local/www/cgi-bin
# chown www:www /usr/local/www/cgi-bin/viralator.cgi
```

Но это еще не все. Для работы viralator нужны модули Perl:

```
# perl -MCPAN -e shell
```

Данная команда будет задавать много вопросов, на которые нужно просто отвечать нажатием Enter — ничего вводить не нужно. После этого появится приглашение:

```
cpan>
```

Введите команду:

```
install LWP
```

Этим вы загрузите и установите модуль LWP. Введите команду quit, после чего введите команды:

```
# mkdir /etc/viralator
# mv -v /usr/viralator-X.X.X/etc/viralator/languages /etc/viralator
```

Первая команда создает конфигурационный каталог для viralator, вторая — помещает в него языковые настройки. Далее нужно создать конфигурационный файл viralator.conf:

```
# ee /etc/viralator.conf
```

Вот содержимое этого файла:

```
servername -> 192.168.1.1
  antivirus -> CLAMAV
  virusscanner -> clamscan
  scannerpath -> /usr/local/bin
  viruscmd -> --remove
  alert -> FOUND
  downloads -> /usr/local/www/downloads
  downloadsdir -> /downloads
  default_language -> english.txt
  secret -> secretpasswd
  scannersummary -> true
  popupfast -> false
  popupback -> false
  popupwidth -> 600
  popupheight -> 400
  filechmod -> 644
  BAR -> bar.png
  PROGRESS -> progress.png
```

Вам нужно изменить только адрес веб-сервера. Сохраните файл конфигурации и создайте упомянутый в нем каталог:

```
# mkdir /usr/local/www/downloads
# chown www:www /usr/local/www/downloads
```

На этом вся настройка завершена. Перезагрузите Squid и Apache, затем перейдите на любую рабочую станцию сети и попробуйте загрузить один из этих файлов:

http://www.eicar.org/download/eicar_com.zip
http://www.eicar.org/download/eicarcom2.zip

Конечно, чтобы все сработало, нужно указать в настройках браузера адрес прокси и порт или же настроить прозрачный прокси-сервер.

Мониторинг сервера

32

32.1. Постановка задачи

Сервер нуждается в постоянном мониторинге, который поможет оптимизировать нагрузку на сервер и правильно распределить его ресурсы. Одним словом, нам нужна система мониторинга системных ресурсов сервера — процессорного времени, памяти, активности сетевых служб, MySQL и т. д. Одной из таких систем является Munin. Система одновременно проста и в то же время предлагает довольно неплохой набор плагинов, позволяющих построить развитую систему мониторинга.

Но не нужно думать, что Munin — единственная система мониторинга. Это не так. Кроме этой системы, возможно, вам захочется использовать системы Zabbix или Nagios. Нельзя сказать, какая из них лучше — они все разные, и выбор системы будет зависеть от поставленных задач. Система Munin больше подходит для мониторинга ресурсов серверов сети, а системы Zabbix и Nagios — для проверки доступности серверов и запущенных на них служб. Подробное описание Zabbix на русском языке вы найдете по адресу:

http://sysadminpages.com/2009/07/freebsd-installing-zabbix/

А с системой Nagios можно познакомиться вот по этому адресу:

http://www.lissyara.su/articles/freebsd/programms/nagios/

В этой книге будет рассмотрена только система Munin.

32.2. Установка и настройка Munin

Система Munin состоит из двух частей: munin-master (серверная часть) и munin-node. Первая устанавливается на сервере и занимается сбором информации, а вторая — устанавливается на клиенте (клиентах). Клиентскую часть можно установить и на сервере, если «мониторить» вы собрались именно его.

Принцип работы системы такой: сбором всей информации с узлов (нодов) занимается сервер — коллектор. На основании собранной информации и строятся красивые графики. Поэтому на все узлы сети, которые вы собрались мониторить, нужно установить munin-node, а на центральный сервер нужно установить munin-master. Если же нужно мониторить и сам центральный узел, то на него тоже нужно установить munin-node.

Первым делом установим первую часть:

```
# cd /usr/ports/sysutils/munin-master/
# make install clean
```

На все вопросы, задаваемые при установке порта, отвечаем y:

```
You need a group "munin".
Would you like me to create it [y]? Y
...
Would you like me to set up log rotation [y]? Y
```

Затем устанавливаем munin-node:

```
# cd /usr/ports/sysutils/munin-node/
# make install clean
```

В файл /etc/rc.conf добавляем следующую строку:

```
munin_node_enable="YES"
```

После этого редактируем файл /usr/local/etc/munin/munin-node.conf. В этом файле нужно указать IP-адреса (указываются в виде регулярных выражений) машин, на которые мы будем загружать графики:

```
allow ^127\.0\.0\.1$
allow ^192\.168\.1\.25$
```

По умолчанию, понятное дело, используется только адрес 127.0.0.1. Запускаем munin-node:

```
/usr/local/etc/rc.d/munin-node start
```

Теперь займемся настройкой munin-master — именно она занимается сбором информации со всех узлов. В файл /usr/local/etc/munin/munin.conf нужно добавить информацию об узлах:

```
[доменное_имя]
address адрес
use_node_name yes
```

Например:

```
[denhost.localdomain]
address 127.0.0.1
use_node_name yes

[host1.localdomain]
address 192.168.1.11
use_node_name yes
```

Также можно отредактировать раздел, где указывается каталог для хранения графиков. Мы будем использовать каталог /usr/local/www/munin, поэтому нужно предоставить пользователю munin необходимые права:

```
# chown -R munin:munin /usr/local/www/munin/
```

Отредактируем расписание планировщика crond для пользователя munin. Сначала нужно войти под именем этого пользователя (что проще сделать с помощью команды su), затем вызвать команду crontab -e для редактирования расписания:

```
# su munin
$ crontab -e
```

В расписание нужно добавить следующие строки

```
MAILTO=root
*/5 * * * * /usr/local/bin/munin-cron
```

Сохраните расписание и введите команду:

```
exit
```

В файл конфигурации Apache (httpd.conf) нужно добавить следующие строки:

```
Alias /stat "/usr/local/www/munin/"
<Directory "/usr/local/www/munin">
  Options -Indexes MultiViews FollowSymlinks
  AllowOverride None
  Order allow,deny
  Allow from all
</Directory>
```

После чего нужно перезапустить Apache.

32.3. Установка плагинов Munin

По большому счету система Munin уже настроена. Осталось только установить необходимые нам плагины — именно они и определяют, какие графики мы получим. Например, плагин cpu «рисует» график загрузки процессора, а плагин mysql — график загрузки сервера MySQL.

Возможные плагины хранятся в каталоге /usr/local/share/munin/plugins/. Например, плагин cpu называется /usr/local/share/munin/plugins/cpu. Для его подключения нужно создать символическую ссылку на плагин в каталоге /usr/local/etc/munin/plugins:

```
# cd /usr/local/etc/munin/plugins
# ln -s /usr/local/share/munin/plugins/cpu cpu
```

Для некоторых плагинов достаточно создать символическую ссылку, а некоторым плагинам этого мало и их нужно настраивать. Установка параметров плагинов производится в файле /usr/local/etc/munin/plugin-conf.d/plugins.conf. Например, для модуля mysql нужно добавить в этот файл строки:

```
[mysql*]
env.mysqlopts -u root -p пароль_пользователя_root
```

```
env.mysqladmin /usr/local/bin/mysqladmin
env.mysqlshow /usr/local/bin/mysqlshow
```

Почти все. Система Munin написана на Perl. Некоторым плагинам нужен пакет DateTime-HiRes, установим его:

```
# cd /usr/ports/devel/p5-DateTime-HiRes
# make install
```

Дополнительные плагины вы можете найти на сайте:

http://exchange.munin-monitoring.org/

Обеспечение безопасности сервера

33

33.1. Защита веб-сервера

Сервер Apache является довольно надежным и защищенным сервером, однако, произведя несколько несложных настроек, мы можем сделать его еще безопаснее. Что делает злоумышленник перед взломом сервера? Он старается собрать больше информации о сервере. Первым делом собирается информация о запущенных сервисах — нужно определить версии программного обеспечения, установленного на сервере, чтобы выбрать стратегию для взлома.

Откройте конфигурационный файл Apache (его месторасположение и название зависит от используемой операционной системы и версии Apache, см. главу 26) и добавьте в него два параметра:

```
ServerSignature Off
ServerTokens Prod
```

Первый параметр управляет отображением подписи Apache (содержащей, как правило, версию Apache) внизу служебных страниц (таких как страницы с сообщениями об ошибке). По таким страницам легко определить версию Apache, что крайне нежелательно.

Второй параметр управляет отображением версии в HTTP-заголовках. Если указать значение `Prod`, то вместо расширенной «подписи» Apache мы увидим только строку «Server: Apache». Версия указана не будет.

Далее нужно внимательно просмотреть конфигурационный файл Apache: нужно найти неиспользуемые модули. Модули загружаются директивой `LoadModule`. Отключив ненужные модули, вы не только обезопасите Apache (чем меньше окон и дверей, тем сложнее попасть в дом), но и сделаете его запуск быстрее. Вам придется изучить, для чего используется тот или иной модуль, и решить, нужен ли он вам. Обычно можно отключить следующие модули, которые в большинстве случаев не нужны: `mod_imap`, `mod_info`, `mod_userdir`, `mod_status`, `mod_cgi`, `mod_autoindex`.

Модуль `mod_userdir` обеспечивает поддержку пользовательских каталогов, а `mod_cgi` — поддержку CGI. Если эти возможности вам нужны, не отключайте данные модули.

Предотвратить DoS-атаку (атаку на отказ) можно путем уменьшения значения директивы Timeout. По умолчанию тайм-аут равен 300 секунд, мы можем без потери функциональности уменьшить это значение до 50 секунд:

```
Timeout 50
```

Осталось защитить файл конфигурации от редактирования и несанкционированного изменения:

```
# chattr +i httpd2.conf
```

Когда вам лично понадобится изменить этот файл, снимите флаг i:

```
# chattr -i httpd2.conf
```

33.2. Защита DNS-сервера

Теперь можно приступить к защите DNS-сервера. Как мы знаем, DNS-серверы обмениваются информацией о зоне: на первичном сервере хранятся оригинальные файлы зоны (которые редактируются вручную администратором), а на вторичном сервере содержится копия зоны, на случай, если первичный сервер выйдет из строя или будет сильно загружен.

Настоятельно рекомендуется определить, кому разрешено передавать информацию о зоне. Для этого в конфигурационный файл DNS-сервера нужно добавить директиву (в директиву options), управляющую передачей (трансфером) зоны:

```
allow-transfer { 192.168.1.2; };
```

192.168.1.2 — это IP-адрес вторичного сервера DNS. Наш первичный DNS-сервер будет передавать информацию о зоне только компьютеру с IP-адресом 192.168.1.2.

Теоретически злоумышленник может вывести из строя вторичный сервер, а пока его будут восстанавливать, он запустит собственный DNS-сервер с IP-адресом 192.168.1.2, получит информацию о зоне, выведет из строя сервер 192.168.1.1 (первичный DNS-сервер) и получит полный контроль над зоной. Все DNS-запросы будут передаваться на «фальшивый» DNS-сервер.

Чтобы такая ситуация не произошла на практике, используется механизм подписи транзакций (Transaction SIGnatures, TSIG). При использовании TSIG информация о зоне будет передана только в том случае, если совпадает ключ — перед передачей зоны производится проверка секретного ключа, который должен быть на первичном и на вторичном сервере DNS. Без этого ключа первичный сервер не передаст информацию на вторичный, а вторичный не получит информацию от первичного. Ведь возможна и другая ситуация, когда злоумышленник выведет из строя первичный сервер DNS и попытается передать «подделанную» зону на вторичный DNS-сервер.

Чтобы защитить наши серверы DNS, нужно сначала создать ключ, который вы потом «пропишете» в конфигурационном файле каждого DNS-сервера. Введите команду:

```
# dnssec-keygen -a hmac-md5 -b 128 -n HOST dns1-dns2
```

Вы увидите результат выполнения команды:

```
Kdns1-dns2.код
```

Кроме этого программа dnssec-keygen создаст два файла Kdns1-dns2.код.key и dns2.код.private. Откройте второй файл в любом текстовом редакторе, в нем будут строки:

```
Private-key-format: v1.2
Algorithm: 157 (HMAC_MD5)
Key: ключ
```

Скопируйте ключ в буфер обмена. Откройте файл конфигурации (named.conf) первичного DNS-сервера и добавьте в самое его начало строки:

```
key dns1-dns2 {
  algorithm hmac-md5;
  secret "ключ";
};
```

Точно такие же строки нужно добавить в конфигурационный файл вторичного DNS-сервера. Далее настройка первичного и вторичного сервера отличается, поэтому будьте внимательны. Начнем с первичного сервера. В его конфигурационный файл нужно добавить строки:

```
server 192.168.1.2  {
key { dns1-dns2; };
};
```

Измените только IP-адрес вторичного сервера DNS. Затем в директиву options нужно добавить директиву ограничения трансфера зоны:

```
options {
…
allow-transfer { 192.168.1.2; };
}
```

Этим мы ограничили передачу зоны только серверу с IP-адресом 192.168.1.2 и указали, что при обмене зоной серверы должны использовать ключ dns1-dns2.

Сохраните файл конфигурации первичного сервера DNS и откройте named.conf на вторичном сервере. В него нужно добавить строки:

```
server 192.168.1.1  {
key { dns1-dns2; };
};
```

Для получения зоны от сервера 192.168.1.1 нужно использовать ключ dns1-dns2. Затем нужно добавить в директиву options директиву, запрещающую передачу зоны:

```
allow-transfer { none; };
```

Сохраните файл конфигурации и перезагрузите оба сервера.

33.3. Защита сервиса Samba

Очень часто сервис Samba настраивается так, что доступ к нему разрешен всем желающим. В небольшой домашней сети, где нужно «расшарить» общую папку с фильмами и музыкой, такое решение — то, что нужно. Но совсем другое дело — сеть в офисе, где нужно разграничить права доступа.

Для домашней сети параметр security можно установить в share. В корпоративной среде нужно использовать значения user или server. Значение server подразумевает использование сервера для аутентификации пользователей — это оптимальное решение в крупной сети. Но если сеть недостаточно крупная или пока нет желания настраивать такой сервер, нужно использовать значение user. В этом случае считается, что имя пользователя и пароль для доступа к ресурсам Samba нужно брать на этом компьютере. Для добавления пользователя нужно использовать обычную команду adduser. В Linux нужно указать оболочку /bin/false (чтобы отключить традиционный вход пользователя в систему):

```
# adduser -s /bin/false samba_user
```

Во FreeBSD можно вызвать adduser без параметров, а в качестве оболочки указать (когда будет запрошено) команду /sbin/nologin.

После этого нужно изменить системный и Samba-пароли:

```
# passwd samba_user
# smbpasswd samba_user
```

Осталось только открыть smb.conf и в секции global изменить параметр security:

```
security = user
```

После этого перезапустите Samba.

33.4. Защита FTP-сервера

В этой книге была рассмотрена настройка FTP-сервера ProFTPD. Данный сервер является одним из самых защищенных, и особых нареканий он никогда не вызывал, если не считать корявых настроек, которые могут допустить некоторые администраторы.

Для большей защищенности нужно в конфигурационном файле сервера установить следующие директивы так:

```
DefaultRoot ~
RequireValidShell on
```

Первая устанавливает корневой каталог для вошедшего по FTP пользователя. Тильда (~) означает домашний каталог, следовательно, когда пользователь войдет по FTP, корневым для него станет его домашний каталог и он не сможет переместиться выше по иерархии файловой системы.

Вторая директива запрещает использовать оболочки (при входе по FTP), не указанные в /etc/shells.

Автоматизация задач. Командный интерпретатор bash

34

34.1. Зачем это нужно?

С помощью командного интерпретатора `bash` мы можем создавать небольшие сценарии, позволяющие автоматизировать выполнение некоторых задач[1].

Сценарий — это обычный текстовый файл, содержащий инструкции, которые должен выполнить командный интерпретатор.

Спрашивается, а зачем нужны сценарии? С помощью сценариев вы сможете намного быстрее выполнять любые действия, например для подключения к Интернету вы каждый раз вводите команду:

```
sudo pon dsl-provider
```

Вы можете создать сценарий `i`, который будет содержать эту команду, и для подключения к Интернету вам нужно будет ввести всего лишь:

```
i
```

Конечно, данный пример тривиален и его можно было бы реализовать с помощью алиасов (псевдонимов) команд, но все же.

А в конце главы мы напишем несколько сценариев, которые пригодятся любому UNIX-пользователю.

34.2. Первый сценарий

Давайте напишем наш первый сценарий. Он будет выглядеть так:

Листинг 34.1. Самый первый сценарий — файл first

```
#!/bin/bash
# Выводим строку
echo "Hello world"
```

[1] И не только с помощью него — на самом деле, можно создавать сценарии вообще на любых интерпретируемых языках программирования (например, perl или python), лишь бы в системе был установлен соответствующий интерпретатор и он мог принимать файл с программой в качестве первого аргумента.

Что с ним делать? Просто запустите gedit, введите приведенные строки и сохраните файл под именем first.

ПРИМЕЧАНИЕ

Если вы пишете сценарий для FreeBSD, то нужно указать следующий путь к интерпретатору bash: /usr/local/bin/bash.

Чтобы запустить файл, нужно присвоить ему право на выполнение. Но не спешите этого делать. Сначала разберемся, что мы написали.

Первая строка — это указание командному интерпретатору, какую программу запустить для выполнения сценария. Мы будем передавать наш сценарий программе /bin/bash — это и есть командный интерпретатор bash.

Вторая строка — это обычный комментарий. Обратите внимание: в первой строке после решетки не должно быть пробела, иначе командный интерпретатор посчитает инструкцию ! (указание программы) за обычный комментарий.

Третья строка — вызов оператора echo, выводящего текст в кавычках — всем известный текст "Hello World"

Теперь сделаем файл исполнимым:

```
chmod +x first
```

Нам осталось только запустить наш сценарий:

```
./first
```

Просто first ввести нельзя, нужно указать командному интерпретатору, что он находится в текущем каталоге, иначе будет произведен поиск сценария first в каталогах, указанных в переменной окружения PATH (/bin, /usr/bin и др.), а его там не будет.

Сценарий выведет строку:

```
Hello world
```

Весь процесс разработки сценария изображен на рис. 34.1 — вызов gedit, вывод созданного сценария, установка права выполнения и вызов сценария

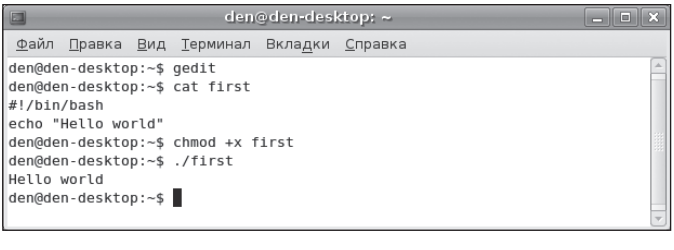

Рис. 34.1. Создание и запуск первого сценария

34.3. Переменные и переменные окружения

В нашем сценарии мы можем использовать переменные, как и в обычной программе. Без переменных никак, иначе где бы мы хранили все промежуточные значения, необходимые для работы сценария?

Переменные объявляются так:

```
имя_переменной=значение
```

До и после знака равенства не должно быть пробелов. Например:

```
NAME="Denis"echo $NAME
```

Значение переменной можно ввести с клавиатуры:

```
echo -n "What is your name? "
read NAME
echo $NAME
```

Параметр `-n` оператора `echo` нужен для того, чтобы после вывода строки он не выводил символ новой строки. Чтение значения переменной осуществляется оператором `read`.

В табл. 34.1 перечислены специальные (служебные) переменные интерпретатора `bash`.

Таблица 34.1. Специальные переменные

Переменная	Значение
$0	Имя сценария
$n	Содержит значение параметра с номером n. При вызове сценария мы можем передать ему параметры, например, ./first par1 par2 Переменная $1 будет содержать строку "par1", а переменная $2 — строку "par2"
$#	Количество параметров, которые были переданы сценарию
$?	Код завершения последний инструкции
$$	PID текущего процесса

Теперь поговорим о переменных окружения (environment variables). Переменные окружения — это глобальные переменные командного интерпретатора, содержащие служебные данные. Наиболее полезные переменные окружения приведены в табл 34.2.

Таблица 34.2. Некоторые переменные окружения

Переменная	Значение
PWD	Текущий каталог
UID	Идентификатор пользователя, запустившего сценарий
HOME	Домашний каталог пользователя, запустившего сценарий
PATH	Путь, по которому производится поиск программ. В Ubuntu по умолчанию он равен /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games
RANDOM	Случайное число от 0 до 32767
REPLY	Последняя строка, введенная с помощью read

Вывести значение переменной окружения можно, как и любой другой переменной, например:

```
echo $PATH
```

Вы можете сами установить переменные окружения с помощью оператора export:

```
export переменная
```

Все переменные, экспортированные с помощью export, будут доступны во всех процессах, порожденных вашим сценарием.

В bash вы также можете использовать массивы. Инициализация массива выглядит так:

```
имя[индекс]=значение

A[0]=1
A[1]=2
```

Обратиться к любому элементу массива можно так:

```
$имя[индекс]
```

Например:

```
echo $A[0]
```

34.4. Подстановка команд

В переменную можно записать вывод любой команды. Что потом с ним делать? Можно изменить, можно просто вывести.

Для подстановки вывода команды используется следующая конструкция (обратите внимание: это не одинарные кавычки, это обратные кавычки):

```
переменная=`команда`
```

Например:

```
CURRENT_DATE=`date`
UPTIME=`uptime`
HOSTNAME=`hostname`
```

34.5. Условные операторы if и case

Рассмотрим синтаксис оператора if:

```
if условие1 then
    список1
elif условие2 then
    список2
else
    список3
fi
```

Работает данный оператор так: если первое условие истинно, то выполняется первый список команд. Если первое условие ложно, то проверяется второе условие. Если оно истинно, выполняется второй список команд. Если и второе условие ложно, тогда выполняется третий список команд.

ПРИМЕЧАНИЕ

Количество блоков elif не ограничено.

Понятно, что вы можете сокращать оператор `if` — совсем не обязательно использовать его полную форму. Вот пример сокращенной формы:

```
if условие1 then
    список1
```

Условия обычно записываются в квадратных скобках — `[]`. Вот пример условия:

```
# строки равны
[str1==str2]
```

Символ ! используется для инвертирования логического выражения (по сути, это операция отрицания NOT), например:

```
# строки не равны
[str1!=str2]
```

В блоке условий вы можете использовать специальные выражения для сравнения значений и проверки существования файла/каталога (табл. 34.3).

Таблица 34.3. Специальные выражения для формирования условий

Условие	Описание
-d <имя>	Истина, если указанное имя является именем каталога
-e <файл>	Истина, если файл существует
-L <файл>	Истина, если файл существует и является ссылкой
-s <файл>	Истина, если файл существует и не пустой (размер больше 0)
-x <файл>	Истина, если файл существует и является исполнимым
-w <файл>	Истина, если файл существует и доступен для записи
-z переменная	Истина, если строка, содержащаяся в переменной, является пустой (ее длина равна 0)
-n переменная	Истина, если строка, содержащаяся в переменной, не является пустой (ее длина больше 0)
переменная -eq значение	Равно
переменная -ne значение	Не равно
переменная -lt значение	Меньше
переменная -gt значение	Больше
переменная -le значение	Меньше или равно
переменная -ge значение	Больше или равно

В выражениях `-eq .. -ge` вместо значения можно указывать имя переменной, с которой будет производиться сравнение.

Вот пример небольшого сценария, использующего оператор сравнения.

Листинг 34.2. Второй сценарий

```
#!/bin/bash
echo -n "В каком году закончилась ВОВ?"
```

```
read year
if [ $year -eq 1945 ]
then echo "Молодец! Знаем историю!"
else echo "Мда... не угадал"
fi
```

Теперь рассмотрим синтаксис второго условного оператора — case:

```
case переменная in
значение1) список1 ;;
...
значениеN) списокN ;;
*) список_по_умолчанию;;
esac
```

Как работает данный оператор, надеюсь, понятно. Если значение переменной совпадает с одним из предполагаемых значений, то будет выполнен соответствующий значению список команд. В противном случае будет выполнен список команд по умолчанию.

Давайте перепишем второй сценарий с помощью case (листинг 34.3).

Листинг 34.3. Сценарий с использованием оператора case

```
#!/bin/bash
echo -n "В каком году закончилась ВОВ?"
read year

case $year in
1945) echo "Молодец! Знаем историю!";;
*) echo "Мда... Не угадал"
esac
```

34.6. Циклы

Интерпретатор bash поддерживает четыре типа циклов — for, while, until и select, но в этой книге мы рассмотрим только два — for и while (они используются чаще всего).

Вот синтаксист цикла for:

```
for переменная in список1
do
список2
done
```

Самый тривиальный пример:

```
for num in 1 2 3 4 5 6 7; do echo -n $num; done
```

В терминале вы увидите:

```
1234567
```

Вот пример посложнее — построчный вывод файлов:

```
for f in `cat report.txt`
do echo $f
done
```

Как видите, цикл for закончит работу, как только будет обработан последний элемент первого списка.

Цикл while выглядит так:

```
while список1
do
список2
done
```

Вот небольшой пример использования данного цикла:

```
i=1
while [ $i -lt 10 ]
do
  echo -n "$i "
  i=$(( $i+1 ))
done
```

В окне терминала вы увидите следующий вывод:

```
1 2 3 4 5 6 7 8 9
```

Цикл завершит свою работу, как только переменная $i достигнет значения 10.

34.7. Полезные сценарии

34.7.1. Сценарий автоматической смены обоев рабочего стола GNOME

Вот теперь мы готовы к тому, чтобы написать собственный сценарий. Нам нужно написать сценарий, который был бы полезен всем пользователям GNOME. Предлагаю написать сценарий автоматической смены обоев рабочего стола. Предположим, что у нас в домашнем каталоге есть каталог pics, содержащий много картинок. Идея заключается в том, чтобы написать сценарий, который бы при запуске брал из данного каталога произвольную картинку и устанавливал в качестве обоев рабочего стола. Потом данный сценарий нужно прописать в автозапуск, и при каждом входе в систему на рабочем столе будет новая картинка. Полезно? Я тоже так думаю.

В листинге 34.4 приведен уже готовый сценарий. Поскольку вы уже знакомы с синтаксисом bash, комментировать его не буду. Единственное, что, возможно, придется изменить — это имя каталога с картинками.

Листинг 34.4. Сценарий автоматической смены обоев рабочего стола

```
#!/bin/bash
export DIR='/home/den/pics/'
export NUM=$RANDOM

export TOTAL=0
for f in `ls $DIR`
```

```
do
    let "TOTAL += 1"
done

let "NUM %= TOTAL"
export CURRENT=0

for f in `ls $DIR`
do
    if [ $CURRENT = $NUM ]
    then
   /usr/bin/gconftool-2 -t string -s      /desktop/gnome/background/picture_filename $DIR/$f
    break
    fi
let "CURRENT += 1"
done
```

ПРИМЕЧАНИЕ

Данный сценарий будет работать только в Ubuntu, точнее в любом дистрибутиве, но при условии, что вы используете графическую среду GNOME. В дистрибутивах, где используется KDE, данный сценарий работать не будет.

Сделаем сценарий исполнимым:

```
chmod +x auto_wallpapers
```

Перед запуском не забудьте поместить в каталог с картинками сами картинки.

Чтобы сценарий запускался автоматически, выполните команду Система ▸ Параметры ▸ Сеансы. В появившемся окне нажмите кнопку Создать. Введите описание программы, например Автоматическая смена обоев, и полное имя сценария, например /home/<имя_пользователя>/auto_wallpapers (рис. 34.2).

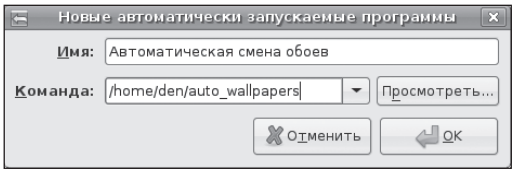

Рис. 34.2. Добавление программы в автозапуск

34.7.2. Мониторинг системного журнала

Предыдущий сценарий полезен для рабочей станции и для домашнего компьютера, а сейчас мы напишем сценарий, полезный для администратора. При отладке различных сетевых служб полезно производить мониторинг системного журнала. Например, когда отлаживается работа VPN-сервера, очень удобно наблюдать, как регистрируются пользователи.

Сейчас мы напишем сценарий, выводящий последние 5 строк из системного журнала /var/log/messages каждые две секунды. Идея очень проста: с помощью

команды tail в бесконечном цикле выводим последние 5 строк (количество строк можно изменить параметром -n), затем ждем две секунды и повторяем итерацию. Завершить работу сценария можно, нажав Ctrl + C. Код сценария приведен в листинге 34.5.

Листинг 34.5. Сценарий мониторинга системного журнала

```
#!/usr/local/bin/bash
# Время ожидания, в секундах
INT=2
echo "Нажмите Ctrl + C для завершения работы сценария"

while [ true ]
do
tail -n 5 /var/log/messages

sleep $INT
# Выводим две пустые строки для большего удобства
echo; echo
done
```

34.7.3. Создание файла подкачки во FreeBSD

Нельзя сказать, что вы будете создавать файл подкачки каждый день, скорее всего, вы создадите его один раз и забудете о нем. Но процесс создания файла подкачки можно автоматизировать, и если у вас несколько серверов, разработанный сценарий mkswap вам пригодится. Даже если вы будете создавать файл подкачки вручную, данный сценарий подскажет вам правильную последовательность действий. Сценарий создания файла подкачки представлен в листинге 34.6. Еще раз отмечу, что данный сценарий будет работать только во FreeBSD.

Листинг 34.6. Сценарий mkswap

```
#!/usr/local/bin/bash
# Для создания и активации файла подкачки нужны права root
# root – это пользователь с UID 0. Поэтому при запуске сценария
# первым делом нужно проверить, равен ли UID пользователя,
# запустившего сценарий, нулю. Если это не так, мы должны завершить
# работу сценария

if [ "$UID" -ne "0" ]
then
echo "Запустите сценарий от имени root"
# 65 – код ошибки, сообщающий о нарушении прав доступа
exit 65
fi


# Имя файла подкачки – измените по своему усмотрению
FILE=/swap-file

# Размер блока (1024 Кб, то есть 1 Мб)
```

```
BLOCKSIZE="1024k"
# Количество блоков
BLOCKS=512

# Размер файла равен BLOCKSIZE x BLOCKS
# Данный сценарий создает файл подкачки размером 512 Мб,
# поэтому убедитесь, что на диске достаточно места

echo "Создаем файл подкачки. Подождите..."

# Команда dd создает пустой файл нужного размера
dd if=/dev/zero of=$FILE bs=$BLOCKSIZE count=$BLOCKS

# Изменяем права доступа к файлу
chmod 0600 $FILE

# Превращаем пустой файл в файл подкачки
mdconfig -a -t vnode -f $FILE -u 0

# Активируем файл подкачки
swapon $FILE

echo "Операция завершена успешно"

# код 0 - успешное завершение сценария
exit 0
```

Чтобы после перезагрузки не пришлось активировать файл подкачки снова, добавьте в файл /etc/rc.conf строку:

```
swapfile="/swap-file"
```

Заключение

Если после прочтения этой книги у вас возникнут вопросы, загляните на форум моего сайта www.dkws.org.ua. На форуме вы найдете ответы на интересующие вас вопросы, касающиеся настройки Linux и UNIX. Также не забывайте о приложениях, в которых рассматривается настройка совместной загрузки FreeBSD, Linux и Windows, процесс обновления Linux и FreeBSD, а также установка FreeBSD с флешки или по сети.

Приложения

- Приложение А. Двойная загрузка — Windows и FreeBSD/Linux
- Приложение Б. Совместная загрузка Linux и FreeBSD
- Приложение В. Обновление Linux
- Приложение Г. Обновление FreeBSD
- Приложение Д. Установка FreeBSD с флешки
- Приложение Е. Установка FreeBSD по сети

Двойная загрузка — Windows и FreeBSD/Linux

Даже не знаю, кому может понадобиться такая конфигурация: Windows и FreeBSD на одном компьютере. Единственное предположение — эксперименты с FreeBSD на домашнем компьютере. Вы можете установить FreeBSD на домашний компьютер, поскольку бюджет обычно не позволяет для экспериментов покупать еще одну машину. А отказываться от Windows дома как-то не хочется — какие бы ни были хорошие эмуляторы Windows, в родном окружении игры работают быстрее и с меньшим количеством глюков.

В этой книге будет рассматриваться совместная загрузка FreeBSD и последних версий операционной системы от Microsoft — Vista и Windows 7. Рекомендации по укрощению старого загрузчика операционных систем Windows 2000 и Windows XP вы без проблем найдете в Интернете — все они сводятся к редактированию файла `boot.ini` и копированию загрузочного сектора FreeBSD.

А вот начиная с Vista, привычного загрузочного файла `boot.ini` уже нет и для редактирования параметров загрузчика придется использовать утилиту `bcdedit`. В ней нет ничего страшного, однако должен отметить, что редактирование `boot.ini` было более простой задачей.

Сначала предположим, что на компьютере еще не установлена операционная система — ни Windows, ни FreeBSD. Первой нужно установить FreeBSD. Установите ее как обычно, загрузчик можно тоже выбрать любой — или стандартный или более продвинутый BootMgr — какой вам больше нравится. Кстати, если вы хотите совместно установить Windows и Linux, то первой нужно устанавливать Windows, а не Linux. Программа установки Linux найдет уже установленную Windows и настроит соответствующим образом загрузчик — вам ничего не придется делать, что не может не радовать. Именно поэтому в данном приложении особое внимание уделяется FreeBSD — нужно будет выполнить часть работы вручную; программа установки FreeBSD, к сожалению, не такая интеллектуальная, как инсталлятор Linux.

После установки FreeBSD войдите в систему как root и скопируйте на флешку файл `/boot/boot1`. Перезагрузите компьютер и установите Windows. После установки Windows запустить FreeBSD уже не получится — не волнуйтесь, все под контролем. Но скопировать файл `/boot/boot1` нужно именно до установки

Windows, иначе потом придется использовать LiveCD, чтобы добраться до нашего /boot/boot1.

Загрузитесь в Windows и скопируйте файл boot1 с флешки в корневой каталог диска C:, назовем его unix.mbr.

Теперь запустите cmd.exe с правами администратора (Пуск ▸ Все программы ▸ Стандартные, правый щелчок на ярлыке Командная строка — дальше сами разберетесь) и введите команду:

```
bcdedit /create /d "My FreeBSD 9" /application bootsector
```

Текстовую строку можете отредактировать по своему усмотрению. Данная команда создаст загрузочную запись и выведет ее идентификатор:

```
The entry {XXXXXXXX-XXXX-XXX-XXX-XXXXXXXXXXXX} was successfully created.
```

Скопируйте идентификатор в буфер обмена и введите следующие команды:

```
bcdedit /set {XXXXXXXX-XXXX-XXX-XXX-XXXXXXXXXXXX}} device boot
bcdedit /set {XXXXXXXX-XXXX-XXX-XXX-XXXXXXXXXXXX} path \unix.mbr
bcdedit /set {XXXXXXXX-XXXX-XXX-XXX-XXXXXXXXXXXX} device partition=c:
bcdedit /displayorder {XXXXXXXX-XXXX-XXX-XXX-XXXXXXXXXXXX} /addlast
```

Вот и все. Если вы все сделали правильно, при перезагрузке в меню загрузчика Windows появится запись My FreeBSD 9, выбрав которую вы запустите загрузчик FreeBSD, который, в свою очередь, загрузит саму систему.

Немного иная ситуация, если Windows уже установлена. Не хочется ее переустанавливать только из-за установки FreeBSD, точнее даже из-за того, что программа установки FreeBSD ни с кем не считается и не может автоматически определить наличие Windows. В этом случае нужно использовать программу EasyBCD (вы без проблем найдете ее в Интернете — она бесплатная).

При установке FreeBSD нужно отказаться от установки загрузчика (выбрать None в соответствующем окне), после установки FreeBSD загрузить Windows, запустить EasyBCD, перейти на вкладку Linux и добавить FreeBSD в меню загрузчика Windows. Данная программа не всегда корректно работает, поэтому первый способ более надежный, но для большинства пользователей не очень удобный.

Совместная загрузка Linux и FreeBSD

Две установленные операционные системы UNIX — это не что-то из ряда фантастики, а вполне нормальная ситуация для экспериментальной площадки. Помню, когда на моем компьютере было установлено пять разных дистрибутивов Linux. Так почему бы не установить Linux вместе с FreeBSD? Хотя в реальных условиях операционная система будет у вас одна — или Linux или FreeBSD. Но пока вы учитесь и до реальных условий еще далеко, нужно организовать совместную загрузку двух похожих, и в то же время совершенно разных операционных систем.

В современных дистрибутивах Linux используются загрузчики GRUB и GRUB2. Ранее использовался LILO, но сейчас он уже не актуален. Поэтому будем ориентироваться на современные загрузчики.

Нужно попытаться установить сначала FreeBSD, а затем Linux. Тут все зависит от программы установки выбранного вами дистрибутива. Одни инсталляторы могут определить FreeBSD и сконфигурировать загрузчик, другие — нет. Но все равно важно установить Linux уже после установки FreeBSD. Во-первых, есть шанс, что все настроится автоматически. Во-вторых, мы же будем настраивать загрузчик Linux, а не FreeBSD, поэтому Linux нужно установить второй системой.

Если при установке Linux первая операционная система (наша FreeBSD) не была обнаружена, придется настраивать загрузчик. Перейдите в каталог `/boot/grub/` и попытайтесь найти файл `menu.lst` — его наличие говорит нам об использовании загрузчика GRUB (наличие файла `grub.cfg` говорит об использовании GRUB2)[1]. Откройте этот файл и добавьте в него строки:

```
title  FreeBSD9
root   (hd0,2)
kernel (hd0,2)/boot/loader
```

[1] Впрочем, возможно именно в используемом вами дистрибутиве это окажется немного не так: например, используется загрузчик GRUB2, но немного модифицированный авторами дистрибутива для поддержки файла menu.lst от предыдущих версий дистрибутива, использовавших GRUB. Поэтому стоит поискать оба файла, даже если вы точно знаете, какая у вас версия загрузчика.

Вам нужно изменить только имя устройства. Мы используем первый жесткий диск и второй раздел.

У загрузчика GRUB есть некоторые проблемы с современной файловой системой UFS2, которая появилась в 5-й версии FreeBSD. Ранние версии GRUB не могли загружать FreeBSD, если использовалась эта файловая система — была поддержка только обычной UFS. Про поддержку экспериментальной ZFS я вообще молчу. Позже были выпущены патчи, позволяющие загружать FreeBSD с UFS2-раздела. Если вы будете использовать современную версию дистрибутива, где до сих пор используется GRUB (например, Mandriva 2010), то велика вероятность, что все необходимые патчи будут применены и проблем с загрузкой FreeBSD не будет.

А вот у более современного загрузчика GRUB2 никаких проблем с UFS2 нет, зато есть проблемы с ZFS, поэтому при установке FreeBSD настоятельно не рекомендуется выбирать ZFS. Вроде бы есть патчи, добавляющие в GRUB2 поддержку ZFS, но лично я их не использовал, поэтому ничего не могу порекомендовать.

Для загрузки FreeBSD средствами GRUB2 нужно добавить в файл /etc/grub.d/40_custom следующие строки:

```
menuentry "FreeBSD 9.0" {
    set root=(hd0,2,a)
    chainloader +1
}
```

Обратите внимание: мы указали первый жесткий диск (/dev/sda в Linux), второй раздел (слайс), имя BSD-раздела — a. После изменения файла /etc/grub.d/40_custom нужно обновить GRUB2:

```
sudo update-grub
```

Обновление Linux

В

Процесс обновления Linux зависит от используемого дистрибутива. В этом приложении мы не будем рассматривать обновление всех дистрибутивов, остановимся только на двух — Ubuntu и Mandriva. Выбор ничем не обосновывается, просто мне так захотелось. Пока могу отметить одно: нужно запускать процесс обновления дистрибутива, а не процесс обновления всех пакетов.

Перед началом обновления системы нужно решить, зачем вам это. Если все работает (все устройства работают, установлены все необходимые программы), то может и не стоит обновляться? Если вам не интересны новые возможности (или их вообще нет, как часто бывает в часто обновляемых дистрибутивах), то все, что изменится, — это номер версии дистрибутива.

Если вам интересны новые возможности, скажем, в новой версии OpenOffice.org, которая появилась в новой версии дистрибутива, то может просто обновить OpenOffice.org? Дело в том, что далеко не всегда обновление проходит гладко. Поэтому перед обновлением системы рекомендуется сделать ее резервную копию с помощью CloneZilla, чтобы после некорректного обновления можно было легко вернуть «все, как было». Иногда проще скопировать все важные для вас данные на другой раздел (или на другой носитель) и установить систему с нуля, а потом вернуть в нее скопированные ранее данные. В любом случае я рекомендую перед началом обновления сделать резервную копию всей системы или хотя бы только всех важных данных.

Начнем с Ubuntu (мы будем рассматривать обновление с 10.04 до 10.10). Существует два способа обновления Ubuntu — простой и неинтересный и более сложный, но более интересный. Первый способ заключается в использовании графического менеджера обновлений. Откройте терминал (или нажмите Alt + F2) и введите команду:

```
sudo update-manager -d
```

Менеджер обновления обновит вашу версию Ubuntu до следующей доступной. А теперь займемся обновлением Ubuntu в консоли — более интересный вариант, заключающийся в вводе команд, — так вы поймете, что делает менеджер обновлений.

Сделаем резервную копию файла sources.list:

```
sudo cp /etc/apt/sources.list /etc/apt/sources.list.bkp
```

Затем нужно в файле sources.list заменить название вашей версии дистрибутива на название версии 10.10 — maverick. Это можно сделать вручную, открыв файл в любом текстовом редакторе, или же с помощью следующей команды:

```
sudo sed -i 's/lucid/maverick/g' /etc/apt/sources.list
```

Мы заменяем в файле sources.list строку 'lucid' на 'maverick'. Запускаем процесс обновления системы:

```
sudo aptitude update
sudo aptitude dist-upgrade
```

Устанавливаем менеджер обновлений и запускаем процесс обновления релиза:

```
sudo aptitude install update-manager-core
sudo do-release-upgrade -d
```

Осталось только перезагрузить компьютер:

```
sudo reboot
```

А вот для обновления Mandriva до версии 2010.1 нужно ввести всего одну команду:

```
# mdkapplet-upgrade-helper --new_distro_version=2010.1
```

Обновление FreeBSD Г

Замечания относительно обновления FreeBSD такие же, как и относительно обновления Linux. Если все прекрасно работает, то может и не стоит ничего обновлять? Не мешайте системе работать — вот основное правило. Тем более FreeBSD в любом случае придется обновлять реже, чем Linux. Ведь FreeBSD в большинстве случаев установлена на сервере. А что такое сервер? Это компьютер, настроенный по принципу «настроил и забыл», «черный ящик» пылящийся в углу (или в специальной стойке — у кого как). А вот рабочая станция (на которой установлена Linux) требует более частого обновления — пользователи требуют более новых программ, да и самому хочется испытать новую версию того же OpenOffice.org или Compiz.

На сервере обновление должно производиться, только если есть прямая необходимость. Нет, я не противник обновления, но обновлять нужно не все подряд (только потому, что вышла новая версия), а только то, что требует обновления. Например, можно обновить старую версию какого-нибудь порта: вышла «заплатка», закрывающая «дыру» в безопасности — порт нужно обновить. Но не будете же вы обновлять всю систему ради нескольких таких заплаток? Совсем другое дело, когда была произведена модернизация системы и потребовалось перейти на новую версию системы ради поддержки нового «железа».

При обновлении FreeBSD осложнения могут возникнуть с большей вероятностью, чем при обновлении Linux. В любом случае перед началом обновления сделайте резервную копию системы. Иногда проще установить и настроить систему с нуля, чем заниматься обновлением системы, особенно с переходом в несколько версий, например с версии 6.0 на версию 8.0. Некоторые эксперты советуют делать поэтапное обновление: с версии 6.0 обновиться хотя бы до 6.3, потом до 7.x, а затем— до 8.0. Но этот процесс довольно-таки сложный и длительный.

Если вы решили обновляться, тогда вам нужно ознакомиться вот с этим документом:

http://www.freebsd.org/releases/8.0R/relnotes-detailed.html

В нем описаны изменения в 8-й версии, в частности можно найти полезные сведения об изменениях в «железе» — некоторые сетевые интерфейсы в 8-й

версии называются иначе по сравнению с 6-й версией. А из-за этого у вас может отказать сеть, да и все сетевые настройки тоже «рухнут», пока вы везде не измените имя интерфейса — это нужно сделать вручную, так как никто за вас не будет редактировать все конфигурационные файлы. Это нужно учесть и отредактировать ваш /etc/rc.conf до первой перезагрузки — особенно это актуально, если вы производите обновление по сети, иначе после перезагрузки вы «потеряете» свой сервер. Когда у вас есть доступ к консоли сервера, ничего страшного нет — вы можете отредактировать rc.conf в любой момент и перезапустить сервер.

Перед тем как обновлять свой сервер, работающий под управлением древней (хотя вполне рабочей) версии 6.2, я ознакомился с множеством тем на самых разных форумах. В большинстве случаев проблемы при обновлении начинались после установки нового «мира», когда отказывались работать базовые команды ee, mv — и даже cp не работала. Проблема заключалась в зависимости от старых библиотек, которые удаляются при обновлении. Чтобы обойти эту проблему, нужно установить порт compat6x из FreeBSD 8 (понятно, что в самой FreeBSD 6 отсутствует порт, реализующий совместимость с версией 6).

И еще один момент. В версии 6.2 впервые появилась утилита freebsd-update, но она настолько стара, что не может выполнить обновление до версии 8.0. Поэтому вам нужно раздобыть ее же, но с другого компьютера, работающего под управление более новой версии FreeBSD, например 7.0. Если же у вас версия 7.x, то все нормально, вы можете использовать родную утилиту freebsd-update.

Начнем обновление. Первым делом нужно установить порт /usr/ports/sysutils/portupgrade, который мы будем использовать для обновления портов:

```
# cd /usr/ports/sysutils/portupgrade
# make install clean
```

Сразу после установки portupgrade обновим порты:

```
# portsnap fetch update
# portupgrade -a
```

Теперь нужно скачать и установить библиотеки, обеспечивающие совместимость с FreeBSD 6:

```
# fetch ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/packages-8.0-release/misc/compat6x-
i386-6.4.604000.200810_3.tbz
# fetch ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/packages-8.0-release/misc/localedata-5-
.4.tbz
# pkg_add compat6x-i386-6.4.604000.200810_3.tbz
# pkg_add localedata-5.4.tbz
```

Первые две команды загружают тарболы compat6x и localedata, а вторые две команды — их устанавливают.

После этого настало время «почистить» наш /etc/rc.conf. Отключаем (комментируем строки) все демоны, оставляем только настройку сети. В строках настройки сети (начинаются с ifconfig_), если нужно, изменяем имена сетевых интерфейсов. Комментируем все строки, которые могут так или иначе помешать нам обновить систему — enable_quota, natd_enable, linux_enable, firewall_enable и т. п. Конфигурационный файл должен быть девственно чист — как после установки системы. Только настройка сети и доменное имя, если нужно (если

не используется DHCP). Также отредактируйте расписание crond — отключите все задания.

Перед запуском процесса обновления рекомендуется удалить каталог /usr/src — все исходные тексты, — так обновление пройдет быстрее. Ведь тогда freebsd-update будет «думать», что исходники просто не установлены, и не будет их обновлять, а это экономия и времени, и трафика. Когда они вам понадобятся, вы всегда можете их взять из дистрибутива новой версии, который вы уже скачали, ведь так? Удаляем каталог /usr/src рекурсивно:

```
# rm -rf /usr/src
```

Вот теперь все готово, и мы можем начать обновление системы:

```
# freebsd-update upgrade -r 8.0-RELEASE
Looking up update.FreeBSD.org mirrors... 3 mirrors found.
Fetching metadata signature for 6.2-RELEASE from update5.FreeBSD.org... done.
Fetching metadata index... done.
Inspecting system... done.

The following components of FreeBSD seem to be installed:
kernel/generic world/base world/catpages world/dict world/doc
world/games world/info world/manpages world/proflibs

The following components of FreeBSD do not seem to be installed:
kernel/smp src/base src/bin src/contrib src/crypto src/etc src/games
src/gnu src/include src/krb5 src/lib src/libexec src/release src/rescue
src/sbin src/secure src/share src/sys src/tools src/ubin src/usbin

Does this look reasonable (y/n)? y

Fetching metadata signature for 8.0-RELEASE from update4.FreeBSD.org... done.
Fetching metadata index... done.
Fetching 1 metadata patches. done.
Applying metadata patches... done.
Fetching 1 metadata files... done.
Inspecting system... done.
Preparing to download files... done.
Fetching 13306 patches.....10....20....30....40....50.....13280....13290....13300...
done.
Applying patches... done.
Fetching 5413 files... done.
```

По возможности freebsd-update автоматически обновит все конфигурационные файлы, например:

```
The following changes, which occurred between FreeBSD 6.2-RELEASE and
FreeBSD 8.0-RELEASE have been merged into <имя файла>:
...
Attempting to automatically merge changes in files... done.
```

Процесс обновления конфигурационных файлов довольно долгий. Программа обновления может предложить новый вариант файла (если не получится обновить автоматически), вы можете принять его или отредактировать в ручном режиме. После будет выведен список файлов, которые в процессе обновления были соответственно удалены, добавлены или обновлены:

```
The following files will be removed as part of updating to 8.0-RELEASE-p0:
The following files will be added as part of updating to 8.0-RELEASE-p0:
The following files will be updated as part of updating to 8.0-RELEASE-p0:
```

Далее нужно обновить ядро. Для этого нам понадобятся новые исходные тексты ядра, которые можно взять из инсталляционного диска, загрузим сам инсталляционный диск и воспользуемся исходниками нового ядра:

```
# cd /root
# fetch ftp://ftp.freebsd.org/pub/FreeBSD/releases/i386/ISO-IMAGES/8.0/8.0-RELEASE-i386-
disc1.iso
# mkdir /mnt/new_bsd
# mount -rt cd9660 /dev/`mdconfig -a -t vnode -f "/root/8.0-RELEASE-i386-disc1.iso"` /mnt/
new_bsd
# cd /mnt/new_bsd/8.0-RELEASE/src
# ./install.sh all
```

Теперь нужно пересобрать ядро. Подробно процесс компиляции ядра в книге не рассматривается — вы без проблем найдете его описание в Интернете. Скажу только, что для сборки ядра нужно ввести вот эти команды:

```
# cd /usr/src
# make buildkernel KERNCONF=GENERIC.mykrnl
# make installkernel KERNCONF=GENERIC.mykrnl
```

В файл `GENERIC.mykrnl` (который создается на базе обычного файла GENERIC) нужно добавить вот такие опции:

```
options IPFIREWALL
options IPFIREWALL_VERBOSE
options IPFIREWALL_VERBOSE_LIMIT=10
options IPFIREWALL_DEFAULT_TO_ACCEPT
options QUOTA
```

Это обычные опции, характерные для любого сервера, — без поддержки межсетевого экрана (брандмауэра) и квот не может обойтись ни один сервер.

Устанавливаем обновление ядра и перезагружаем машину:

```
# freebsd-update install
Installing updates...
Kernel updates have been installed. Please reboot and run
"./freebsd-update install" again to finish installing updates.
# reboot
```

Будем надеяться, что после перезагрузки наш сервер загрузится. Если все нормально, после перезагрузки введите команду:

```
# uname -r
```

Если в выводе вы увидите версию `8.0-RELEASE`, значит, пока все идет по плану. Можно приступить к обновлению мира:

```
# freebsd-update install
Installing updates...
Completing this upgrade requires removing old shared object files.
Please rebuild all installed 3rd party software (e.g., programs
installed from the ports tree) and then run "./freebsd-update install"
again to finish installing updates.
```

На все про все понадобится около 40 минут. После чего нужно перезагрузить компьютер:

```
# reboot
```

Настало время обновить все порты. Да, нужно будет пересобрать все установленные порты. Процесс мучительно долгий. Первым делом нужно пересобрать порт ruby, потому что от него зависит утилита portupgrade:

```
# portupgrade -f ruby
```

Если вы получите сообщение о «битой» базе портов, удалите ее:

```
# rm /var/db/pkg/pkgdb.db
# rm /usr/ports/INDEX-8.db
```

Потом введите команды:

```
# portsdb -fu
# pkgdb -fu
```

Пересоберем все порты:

```
# portupgrade -fa
```

Если вы получите сообщение о наличии неизвестных опций в старых портах, придется пересобрать все порты вручную:

```
# portupgrade -f <имя порта>
```

Осталось последний раз запустить freebsd-update:

```
# freebsd-update install
Installing updates... done.
```

Вот и все. Отредактируйте rc.conf — включите ранее выключенные сервисы и перезагрузите компьютер еще раз:

```
# reboot
```

Обновление до 8-й версии выполнено.

Установка FreeBSD с флешки

В мире Linux загрузочная флешка с инсталляционным дистрибутивом — нормальное явление. Во-первых, она содержит Live-версию самой операционной системы и может использоваться для загрузки «своей» (имеется в виду со своими настройками и программами) операционной системы на произвольном компьютере — лишь бы загрузка с флешки поддерживалась. Принцип «все свое ношу с собой» еще никто не отменял. Во-вторых, загрузочная флешка может использоваться для установки на компьютеры, не оснащенные приводом DVD, — на нетбуки и некоторые серверы, где для удешевления конфигурации DVD-привод не устанавливается. В-третьих, загрузочную флешку удобно использовать для восстановления системы — «брелок» восстановления всегда с вами, нет надобности носить с собой громоздкий DVD-диск.

В случае с FreeBSD мотивы будут примерно теми же. Создав загрузочную флешку, вы можете установить с нее FreeBSD на компьютеры, где отсутствует DVD-привод. Раньше для создания загрузочной флешки во FreeBSD нужно было выполнить ритуал танца с бубном вокруг компьютера. Сейчас все намного проще. На FTP-сервере `freebsd.org` можно скачать файл `8.N-RELEASE-i386-memstick.img`, представляющий собой образ загрузочной флешки, и все, что вам осталось — это записать данный образ на флешку. Для этого вам понадобится компьютер под управлением FreeBSD или Linux, на котором введите команду[1]:

```
# dd if=8.N-RELEASE-i386-memstick.img of=/dev/da0 bs=10240 conv=sync
```

Нужно перейти в каталог с загруженным образом флешки, изменить его имя (`N` в имени файла — это число, соответствующее версии FreeBSD), параметр `of` — это имя флешки. Сразу после подключения флешки введите команду:

```
# tail /var/log/messages | grep kernel
```

[1] Тем не менее, прежде чем, не задумываясь, вводить команду и удивляться весьма возможным некорректным результатам, дочитайте все приложение до конца (включая это примечание), а затем уточните и откорректируйте имена файлов используемого образа и нужной флешки в вашей операционной системе.

Вы увидите среди прочей информации имя устройства флешки:

```
…
Nov 10 10:19:11 denhost kernel: da0 at umass-sim0 bus 0 target 0 lun 0
…
```

В нашем случае файл устройства флешки называется /dev/da0. В Linux, скорее всего, файл устройства будет называться /dev/sdb, при условии, что у вас всего один жесткий диск. После записи образа на флешку нужно загрузиться с нее.

Установка FreeBSD по сети

В предыдущем приложении был рассмотрен способ установки FreeBSD на флешку. Такой вариант установки поможет, если компьютер не оснащен приводом DVD. Но в некоторых случаях (нет под рукой флешки, не поддерживается загрузка с флешки и т. д.) вам придется воспользоваться установкой по сети, поскольку другого способа установки у вас не будет.

Вот только для установки по сети вам понадобится еще один компьютер, выполняющий роль сервера загрузки. Он тоже должен работать под управлением FreeBSD, хотя можно настроить соответствующим образом и Linux, но в этом приложении рассматривается только настройка FreeBSD.

На сервере загрузки нужно установить и настроить сервисы TFTP, NFS и DCHP. DHCP будет выполнять роль BOOTP-сервера, позволяющего загрузить операционную систему по сети. Кстати, если компьютер, на который вы планируете установить систему по сети, не поддерживает загрузку по сети, то даже не знаю, как вы будете устанавливать на него систему (при проблемах с флешкой и отсутствием DVD), наверное, проще установить DVD-привод. Подумайте сами: раз загрузка по сети и с флешки не поддерживается, то компьютер старый, следовательно, гарантии уже нет и можно смело срывать пломбы.

Первым делом настроим TFTP. Откройте /etc/inetd.conf и раскомментируйте строку:

```
tftp dgram udp wait root /usr/libexec/tftpd tftpd -l -s /tftpboot
```

Сохраните изменения. Ничего устанавливать не нужно, так как TFTP обычно установлен по умолчанию. Следующий шаг — настройка сетевой файловой системы NFS. Откройте /etc/exports и добавьте строку (адрес сети нужно изменить):

```
/tftpboot -network 192.168.21 -mask 255.255.255.0
```

В каталог /tftpboot нужно скопировать все содержимое инсталляционного диска FreeBSD. Выполните команды (не забудьте вставить диск в привод):

```
# mount /dev/cdrom
# cp -Rp /mnt/cdrom /tftpboot
```

Еще нужно добавить в /tftpboot/boot/loader.conf строку:

```
vfs.root.mountfrom="ufs:/dev/md0c"
```

Настало время установить DHCP-сервер. Введите команду:

```
# pkg_add -r isc-dhcp30-server
```

Отредактируйте файл /usr/local/etc/dhcpd.conf так:

```
authoritative;
          subnet 192.168.21.0 netmask 255.255.255.0 {
          }
          host instsrv {
            # укажите MAC-адрес "клиента"
     # на этот компьютер мы будем устанавливать ОС
            hardware ethernet 00:11:22:33:44:55;
            # IP-адрес клиента
            fixed-address 192.168.21.54;
            # IP-адрес этого сервера
            next-server 192.168.21.2;
            # Путь к загрузчику
            filename "boot/pxeboot";
            # Корневой каталог TFTP-сервера
            option root-path "/tftpboot";
         }
```

Измените IP-адрес сети, MAC-адрес «клиента», а также IP-адрес «клиента» и сервера. Мы почти все настроили. Остались мелочи. Откройте /etc/rc.conf и добавьте в него строки (точнее, вам придется не просто добавить, а отредактировать, оптимизировав под ваши нужды — хотя бы изменить IP-адрес):

```
# IP-адрес сервера
ifconfig_em0="inet 192.168.21.2 netmask 255.255.255.0"
dhcpd_enable="YES"              # Запускаем DHCP
rpcbind_enable="YES"            # Запускаем RPC (Remote Procedure Call)
nfs_server_enable="YES"         # Запускаем NFS
inetd_enable="YES"              # Запуск суперсервера inetd
```

Сохраняем изменения и перезагружаем компьютер. После его загрузки включаем компьютер, на который нужно установить FreeBSD, и когда начнется загрузка, нужно выбрать тип источника NFS и ввести расположение файлов:

```
192.168.21.2:/tftpboot
```

Денис Николаевич Колисниченко

Администрирование Unix-сервера и Linux-станций

Заведующий редакцией	*А. Кривцов*
Руководитель проекта	*А. Кривцов*
Ведущий редактор	*Ю. Сергиенко*
Редактор	*А. Гущин*
Корректор	*В. Листова*
Верстка	*Л. Харитонов*

ООО «Мир книг», 198206, Санкт-Петербург, Петергофское шоссе, 73, лит. А29.
Налоговая льгота — общероссийский классификатор продукции ОК 005-93, том 2; 95 3005 — литература учебная.
Подписано в печать 24.03.11. Формат 70x100/16. Усл. п. л. 32,250. Тираж 1500. Заказ
Отпечатано по технологии CtP в ООО «Северо-Западный Печатный двор»,
188300, Ленинградская обл., г. Гатчина, ул. Железнодорожная, 45Б

ПРЕДСТАВИТЕЛЬСТВА ИЗДАТЕЛЬСКОГО ДОМА «ПИТЕР»
предлагают эксклюзивный ассортимент компьютерной, медицинской, психологической, экономической и популярной литературы

РОССИЯ

Санкт-Петербург м. «Выборгская», Б. Сампсониевский пр., д. 29а
тел./факс: (812) 703-73-73, 703-73-72; e-mail: sales@piter.com

Москва м. «Электрозаводская», Семеновская наб., д. 2/1, корп. 1, 6-й этаж
тел./факс: (495) 234-38-15, 974-34-50; e-mail: sales@msk.piter.com

Воронеж Ленинский пр., д. 169; тел./факс: (4732) 39-61-70
e-mail: piterctr@comch.ru

Екатеринбург ул. Бебеля, д. 11а; тел./факс: (343) 378-98-41, 378-98-42
e-mail: office@ekat.piter.com

Нижний Новгород ул. Совхозная, д. 13; тел.: (8312) 41-27-31
e-mail: office@nnov.piter.com

Новосибирск ул. Станционная, д. 36; тел.: (383) 363-01-14
факс: (383) 350-19-79; e-mail: sib@nsk.piter.com

Ростов-на-Дону ул. Ульяновская, д. 26; тел.: (863) 269-91-22, 269-91-30
e-mail: piter-ug@rostov.piter.com

Самара ул. Молодогвардейская, д. 33а; офис 223; тел.: (846) 277-89-79
e-mail: pitvolga@samtel.ru

УКРАИНА

Харьков ул. Суздальские ряды, д. 12, офис 10; тел.: (1038057) 751-10-02
758-41-45; факс: (1038057) 712-27-05; e-mail: piter@kharkov.piter.com

Киев Московский пр., д. 6, корп. 1, офис 33; тел.: (1038044) 490-35-69
факс: (1038044) 490-35-68; e-mail: office@kiev.piter.com

БЕЛАРУСЬ

Минск ул. Притыцкого, д. 34, офис 2; тел./факс: (1037517) 201-48-79, 201-48-81
e-mail: gv@minsk.piter.com

Ищем зарубежных партнеров или посредников, имеющих выход на зарубежный рынок.
Телефон для связи: **(812) 703-73-73**. **E-mail:** fuganov@piter.com

---

Издательский дом «Питер» приглашает к сотрудничеству авторов. Обращайтесь по телефонам: **Санкт-Петербург** – **(812) 703-73-72**, **Москва** – **(495) 974-34-50**

Заказ книг для вузов и библиотек по тел.: (812) 703-73-73.
Специальное предложение – e-mail: kozin@piter.com

Заказ книг по почте: на сайте **www.piter.com**; по тел.: (812) 703-73-74
по ICQ 413763617

УВАЖАЕМЫЕ ГОСПОДА!
КНИГИ ИЗДАТЕЛЬСКОГО ДОМА «ПИТЕР»
ВЫ МОЖЕТЕ ПРИОБРЕСТИ
ОПТОМ И В РОЗНИЦУ У НАШИХ
РЕГИОНАЛЬНЫХ ПАРТНЕРОВ.

ДАЛЬНИЙ ВОСТОК

Владивосток
«Приморский торговый дом книги»
тел./факс: (4232) 23-82-12
e-mail: bookbase@mail.primorye.ru

Хабаровск, «Деловая книга», ул. Путевая, д. 1а
тел.: (4212) 36-06-65, 33-95-31
e-mail: dkniga@mail.kht.ru

Хабаровск, «Книжный мир»
тел.: (4212) 32-85-51, факс: (4212) 32-82-50
e-mail: postmaster@worldbooks.kht.ru

Хабаровск, «Мирс»
тел.: (4212) 39-49-60
e-mail: zakaz@booksmirs.ru

ЕВРОПЕЙСКИЕ РЕГИОНЫ РОССИИ

Архангельск, «Дом книги», пл. Ленина, д. 3
тел.: (8182) 65-41-34, 65-38-79
e-mail: marketing@avfkniga.ru

Воронеж, «Амиталь», пл. Ленина, д. 4
тел.: (4732) 26-77-77
http://www.amital.ru

Калининград, «Вестер»,
сеть магазинов «Книги и книжечки»
тел./факс: (4012) 21-56-28, 6 5-65-68
e-mail: nshibkova@vester.ru
http://www.vester.ru

Самара, «Чакона», ТЦ «Фрегат»
Московское шоссе, д.15
тел.: (846) 331-22-33
e-mail: chaconne@chaccone.ru

Саратов, «Читающий Саратов»
пр. Революции, д. 58
тел.: (4732) 51-28-93, 47-00-81
e-mail: manager@kmsvrn.ru

СЕВЕРНЫЙ КАВКАЗ

Ессентуки, «Россы», ул. Октябрьская, 424
тел./факс: (87934) 6-93-09
e-mail: rossy@kmw.ru

СИБИРЬ

Иркутск, «ПродаЛитЪ»
тел.: (3952) 20-09-17, 24-17-77
e-mail: prodalit@irk.ru
http://www.prodalit.irk.ru

Иркутск, «Светлана»
тел./факс: (3952) 25-25-90
e-mail: kkcbooks@bk.ru
http://www.kkcbooks.ru

Красноярск, «Книжный мир»
пр. Мира, д. 86
тел./факс: (3912) 27-39-71
e-mail: book-world@public.krasnet.ru

Новосибирск, «Топ-книга»
тел.: (383) 336-10-26
факс: (383) 336-10-27
e-mail: office@top-kniga.ru
http://www.top-kniga.ru

ТАТАРСТАН

Казань, «Таис»,
сеть магазинов «Дом книги»
тел.: (843) 272-34-55
e-mail: tais@bancorp.ru

УРАЛ

Екатеринбург, ООО «Дом книги»
ул. Антона Валека, д. 12
тел./факс: (343) 358-18-98, 358-14-84
e-mail: domknigi@k66.ru

Екатеринбург, ТЦ «Люмна»
ул. Студенческая, д. 1в
тел./факс: (343) 228-10-70
e-mail: igm@lumna.ru
http://www.lumna.ru

Челябинск, ООО «ИнтерСервис ЛТД»
ул. Артиллерийская, д. 124
тел.: (351) 247-74-03, 247-74-09,
247-74-16
e-mail: zakup@intser.ru
http://www.fkniga.ru, www.intser.ru

ВАМ НРАВЯТСЯ НАШИ КНИГИ? ЗАРАБАТЫВАЙТЕ ВМЕСТЕ С НАМИ!

У Вас есть свой сайт?
Вы ведете блог?
Регулярно общаетесь на форумах? Интересуетесь литературой, любите рекомендовать хорошие книги и хотели бы стать нашим партнером?
ЭТО ВПОЛНЕ РЕАЛЬНО!

СТАНЬТЕ УЧАСТНИКОМ ПАРТНЕРСКОЙ ПРОГРАММЫ ИЗДАТЕЛЬСТВА «ПИТЕР»!

- *Зарегистрируйтесь на нашем сайте в качестве партнера по адресу* ***www.piter.com/ePartners***
- *Получите свой персональный уникальный номер партнера*
- *Выбирайте книги на сайте* ***www.piter.com,*** *размещайте информацию о них на своем сайте, в блоге или на форуме и добавляйте в текст ссылки на эти книги (на сайт* ***www.piter.com****)*

ВНИМАНИЕ! *В каждую ссылку необходимо добавить свой персональный уникальный номер партнера.*

С этого момента получайте 10% от стоимости каждой покупки, которую совершит клиент, придя в интернет-магазин «Питер» по ссылке с Вашим партнерским номером. А если покупатель приобрел не только эту книгу, но и другие издания, Вы получаете дополнительно по 5% от стоимости каждой книги.

Деньги с виртуального счета Вы можете потратить на покупку книг в интернет-магазине издательства «Питер», а также, если сумма будет больше 500 рублей, перевести их на кошелек в системе Яндекс.Деньги или Web.Money.

Пример партнерской ссылки:
http://www.piter.com/book.phtml?978538800282 – обычная ссылка
http://www.piter.com/book.phtml?978538800282**&refer=0000** – партнерская ссылка, где 0000 – это ваш уникальный партнерский номер

Подробно о Партнерской программе ИД «Питер» читайте на сайте WWW.PITER.COM